The expressive power of Memory Logics

Carlos Areces, Diego Figueira, Santiago Figueira, Sergio Mera

The Expressive Power of Memory Logics

Carlos Areces Diego Figueira Santiago Figueira Sergio Mera

May 10, 2011

Abstract

We investigate the expressive power of *memory logics*. These are modal logics extended with the possibility to store (or remove) the current node of evaluation in (or from) a *memory*, and to perform membership tests on the current memory. From this perspective, the hybrid logic $\mathcal{HL}(\downarrow)$, for example, can be thought of as a particular case of a memory logic where the memory is an indexed list of elements of the domain.

This work focuses in the case where the memory is a *set*, and we can test whether the current node belongs to the set or not. We prove that, in terms of expressive power, the memory logics we discuss here lie between the basic modal logic $\mathcal{K}$ and $\mathcal{HL}(\downarrow)$. We show that the satisfiability problem of most of the logics we cover is undecidable. The only logic with a decidable satisfiability problem is obtained by imposing strong constraints on which elements can be memorized.

1 Modal Logics and Memory Logics

Nowadays, the term *modal logics* loosely refers to an extremely wide variety of languages, which are used in many different applications (see, e.g., (Blackburn et al., 2006)). Actually, the fact that the number of members in this family keeps constantly increasing is one of the defining characteristic of the field. While most modal logics have certain general aspects in common (e.g., they are usually interpreted in terms of relational structures and they are computationally well behaved), there usually are as many modal logics satisfying any of these "characterizing properties" as there are modal logics not honoring them. As a result, it is very hard indeed to come up with a proper definition of what a modal logic is. Perhaps one of the few general traits of the field is the desire to investigate languages *specially tailored* for specific tasks.

In this article we investigate the expressive power of a family of modal logics called *memory logics*, which extend both the semantics and the syntax of the classical modal logic. Many logical properties of memory logics have been investigated in recent articles. The original idea was introduced in (Areces, 2007). Areces, Figueira, Gorín, & Mera (2009) investigate tableau algorithms and model checking for memory logics, while Areces,

Figueira, & Mera (2009) discuss axiomatic completeness results. In this article we extend results originally presented in (Areces et al., 2008) and provide full proofs.

We will introduce and motivate memory logics now. We need first some basic definitions. Let $\mathcal{S}$ be a first-order relational signature (i.e., a first order signature without function and constant symbols), and let $\mathcal{M} = \langle D, \mathcal{I} \rangle$ be a relational structure interpreting $\mathcal{S}$ (i.e., D is a non empty set and $\mathcal{I}$ is an interpretation function that assigns to all relational symbols in $\mathcal{S}$ a relation of the correct arity). It is well known that the basic modal language $\mathcal{K}$ can be interpreted on $\mathcal{M}$ (see (Blackburn et al., 2001) for details). When interpreting modal formulas on relational structures, elements in the domain are sometimes called *states*, and the interpretations of relational symbols are called *accessibility relations*.

It is often said that modal languages provide an *internal* perspective of the structures over which they are evaluated. As Blackburn, de Rijke, & Venema (2001) put it,

> "a modal formula [can be seen as] a little automaton standing at some state in a relational structure, and only permitted to explore the structure by making journeys to neighboring states."

It is natural to think of a modal formula as *exploring* the structure, but what about *changing* it? Suppose we want to grant our little automaton the additional power to *modify* the structure during its exploratory trips. This question is not new, and it has resulted in different proposals of what are called *dynamic* logics.

Consider, for example, the task of assigning semantics to a programming language. Clearly, the different instructions of the language change the computational state. It is then natural to define their semantics by specifying which changes each atomic operation of the language introduces. This idea is at the core of formalisms like Hoare-Floyd logics (Floyd, 1967; Hoare, 1969) which include, for example, special operators to indicate the state of variables before and after a given instruction.

As a second example, consider the area of linguistics called dynamic semantics. One of its fundamental claims is that the standard truth-conditional view of sentence meaning —which is the result of using classical logic as representation languages— does not do sufficient justice to the fact that uttering a sentence changes the context it was uttered in. Deriving inspiration, in part, from work on the semantics of programming languages, dynamic semantic theories have developed several variations on the idea that the meaning of a sentence should be equated with the changes it makes to a context. Different dynamic logics like those introduced by Groenendijk & Stokhof in (1991a; 1991b) try to capture these ideas.

As yet a third example with an ample literature, we can mention dynamic epistemic logics (Plaza, 1989; Gerbrandy, 1999; van Benthem, 2001, 2005; van Benthem et al., 2006; van Ditmarsch et al., 2007). These logics model the evolution of the knowledge of epistemic agents via *updates* to the model representing their epistemic state. For example, some of these languages represent the act of an agent updating its epistemic state with the information that φ is true by eliminating all alternative epistemic states where $\neg\varphi$ holds.

Our last family of examples come from the area of temporal logics for verification. In this area, it is many times necessary to model time-critical systems that depend on

quantitative rather than qualitative properties. Many temporal logics introduced for this task use explicit global clocks which are accessed and controlled through logical operators. Examples of such logics are XCTL (Harel et al., 1990), half-order logics (Alur & Henzinger, 1989; Henzinger, 1990), and timed and metric temporal logics (Alur et al., 1993, 1996; Koymans, 1990; Ouaknine & Worrell, 2005).

By contrast, other logics which are also called *dynamic* are not dynamic in the sense mentioned above, the main example being Propositional Dynamic Logic PDL, (Harel, 1984). In PDL formulas are evaluated in a model but they cannot modify it (even though the language does include special operators to verify that certain property holds in a given state and continue evaluation accordingly, which provide extended expressivity (Berman & Paterson, 1981)).

Memory logics can be seen as an attempt to investigate some of the common characteristics of all these logics, in the simplest possible set up. Going back to our little automaton, suppose we extend our definition of a model to a triple $\mathcal{M} = \langle D, \mathcal{I}, M\rangle$, where M is an arbitrary subset of D. We can think of M as a memory where the automaton can store states that are considered particularly interesting. Defining the semantics of this operator is straightforward. Let us write $\langle D, \mathcal{I}, M\rangle, w \models \varphi$ for $w \in D$ and φ a formula to indicate that φ is true at w in the relational structure $\langle D, \mathcal{I}\rangle$ extended with the memory M. Let us use $\textcircled{\texttt{r}}$ ('remember') to represent the memorize operator. We can then define

$$\langle D, \mathcal{I}, M\rangle, w \models \textcircled{\texttt{r}}\varphi \text{ iff } \langle D, \mathcal{I}, M \cup \{w\}\rangle, w \models \varphi.$$

In other words, $\textcircled{\texttt{r}}$ is an instruction to modify the memory of the model, and φ is evaluated in the modified structure. The operation $\textcircled{\texttt{r}}$ by itself is totally useless. If we cannot access the information stored in M, $\textcircled{\texttt{r}}\varphi$ is equivalent to φ. Let us add then an operator $\textcircled{\texttt{k}}$ ('known') that checks whether the current state has been previously remembered:

$$\langle D, \mathcal{I}, M\rangle, w \models \textcircled{\texttt{k}} \text{ iff } w \in M.$$

This simple language gives us already new tautologies. For example, it is easy to see that the formula $\textcircled{\texttt{r}}\textcircled{\texttt{k}}$ is always true. It is also not difficult to see (using well known results from modal logic) that the memory logic operator gives us additional expressivity. Let us remind the semantics of the standard (unary) modal operator diamond $\langle r\rangle$ of the basic modal language[1]. Assuming that $\mathcal{I}(r)$ is a binary relation, we define:

$$\langle D, \mathcal{I}, M\rangle, w \models \langle r\rangle\varphi \text{ iff for some } w' \in D \text{ s.t. } (w, w') \in \mathcal{I}(r)\ \langle D, \mathcal{I}, M\rangle, w' \models \varphi.$$

That is, the formula $\langle r\rangle\varphi$ is true in a state w if the formula φ is true in an r-successor. Now, the memory logic formula $\textcircled{\texttt{r}}\langle r\rangle\textcircled{\texttt{k}}$ is true in a state when evaluated on a model with an empty memory if and only if it is self reachable via the accessibility relation $\mathcal{I}(r)$. I.e.,

$$\langle D, \mathcal{I}, \emptyset\rangle, w \models \textcircled{\texttt{r}}\langle r\rangle\textcircled{\texttt{k}} \text{ iff } (w, w) \in \mathcal{I}(r).$$

[1] Of course, the operator is usually defined on models without memory. We will define it so that it does not interact with the memory M.

As formulas of the basic modal language have the tree model property (i.e., a formula is satisfiable if and only if it is satisfiable in model which is a tree, and hence it does not contain reflexive loops (Blackburn et al., 2001)), this property cannot be expressed in the basic modal language.

In the same spirit of the operators ⓡ and ⓚ introduced above, we can naturally define operators that modify any element of a model (adding or deleting states or modifying the interpretation function). In this paper we will restrict ourselves to operators that can access and modify only the memory M (even though we will briefly discuss possible alternative structures for M). In Section 2 we will formally introduce the syntax and semantics of the memory logics we will investigate. In Section 3 we will define suitable notions of model equivalence for each language, which we will use in Section 4 to investigate their expressive power. In Section 5 we will show that most of the languages obtained, even in this simple set up, are undecidable. We show one case where decidability is regained by imposing a very strict 'memorization policy'. Section 6 finishes the paper with our conclusions and ideas for future work.

We close this section with some additional details on how memory logics were originally conceived, and how they relate to binding and hybrid logics.

Memory Logics and Hybrid Logics, or how Memory Logics were Born

Memory logics where initially defined for purely theoretical reasons (related to questions concerning binding and decidability), but it soon became clear that they could provide an interesting perspective on the question of how a formula can modify the model in which it is being evaluated, as we discussed above.

Memory logics were originally inspired by hybrid logics containing binders like $\mathcal{HL}(\downarrow)$ (see (Areces & ten Cate, 2006)). But while $\downarrow$ was introduced to investigate *dynamic naming* of elements in a model, memory logics include operators to *store* and *retrieve* information from some kind of information structure or memory. In any case, once we take the appropriate point of view $\mathcal{HL}(\downarrow)$ can be considered the first memory logic.

Let us start by formally introducing $\mathcal{HL}(\downarrow)$. Assume a signature $\mathcal{S} = \langle \textsc{prop}, \textsc{nom}, \textsc{rel} \rangle$, where PROP, NOM and REL are countably infinite, pairwise disjoint sets of propositional, nominal and relational symbols respectively. For simplicity, and as it is usually done with modal languages, we will only introduce unary modal operators[2]. The syntax of $\mathcal{HL}(\downarrow)$ is defined as follows

$$\varphi ::= \top \mid p \mid i \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle r \rangle \varphi \mid \downarrow i.\varphi,$$

where $p \in \textsc{prop}$, $i \in \textsc{nom}$ and $r \in \textsc{rel}$. We can see that the language of $\mathcal{HL}(\downarrow)$ is the language of the basic modal logic $\mathcal{K}$ (see (Blackburn et al., 2001) for details) extended with nominals and $\downarrow i$.

[2]Actually, we will restrict ourselves to unary modalities through the article.

Semantically, $\mathcal{HL}(\downarrow)$ is also very close to $\mathcal{K}$. $\mathcal{HL}(\downarrow)$-formulas are interpreted on relational structures extended with an assignment function to interpret nominals. Formally, a model for $\mathcal{HL}(\downarrow)$ is a tuple $\langle D, \mathcal{I}, g\rangle$ where $g : \textsc{nom} \to D$ is an assignment function. $\mathcal{I}$ assigns a subset of D to elements in PROP, and a binary relation on D to elements of REL. Given $\langle D, \mathcal{I}, g\rangle$, the semantic conditions for $\mathcal{HL}(\downarrow)$ are defined as:

$$
\begin{array}{lll}
\langle D,\mathcal{I},g\rangle, w \models \top & \text{iff} & \text{always} \\
\langle D,\mathcal{I},g\rangle, w \models p & \text{iff} & w \in \mathcal{I}(p) \\
\langle D,\mathcal{I},g\rangle, w \models i & \text{iff} & g(i) = w \\
\langle D,\mathcal{I},g\rangle, w \models \neg\varphi & \text{iff} & \langle D,\mathcal{I},g\rangle, w \not\models \varphi \\
\langle D,\mathcal{I},g\rangle, w \models \varphi \wedge \psi & \text{iff} & \langle D,\mathcal{I},g\rangle, w \models \varphi \text{ and } \langle D,\mathcal{I},g\rangle, w \models \psi \\
\langle D,\mathcal{I},g\rangle, w \models \langle r\rangle\varphi & \text{iff} & \text{there is } w' \text{ s.t. } (w,w') \in \mathcal{I}(r) \text{ and } \langle D,\mathcal{I},g\rangle, w' \models \varphi \\
\langle D,\mathcal{I},g\rangle, w \models \downarrow i.\varphi & \text{iff} & \langle D,\mathcal{I},g'\rangle, w \models \varphi \text{ where } g'(j) = g(j) \text{ for } j \neq i \\
 & & \text{and } g'(i) = w.
\end{array}
$$

One way of looking at the semantic condition for $\downarrow i.\varphi$ is that it dynamically creates a name for the current state (by linking the nominal i to it), so that we can later refer to it during the evaluation of φ. An alternative perspective is to see $\downarrow i$ as an instruction to modify the model (by storing the current point of evaluation into i), and continue the evaluation of φ in the modified model. The difference between the two perspectives is subtle, but important for this article. In the latter, we are considering the assignment g as a kind of memory in our model, while $\downarrow i$ and i are the tools we use to access the memory for reading and writing. The question then presents itself naturally: are there other kinds of interesting memory structures and memory operators?

The assignment g is a very sophisticated memory structure: it has unbounded size, it provides direct access to all its memory cells, and each stored element can be unequivocally retrieved. The memory M we discussed above, together with the operators Ⓡ and Ⓚ, provides a much simpler memory structure. Intuitively, these operators cannot discern between different states stored in M, while an assignment g keeps a complete mapping between states and nominals. But notice that Ⓡ *is* a binder, and effectively binds instances of Ⓚ appearing in its scope. In other words, as we can see $\downarrow i$ and nominals as memory operators which store and retrieve information from a memory structure, we could see Ⓡ as a binder that binds occurrences of Ⓚ in its scope. As the memory structure used by Ⓡ and Ⓚ has less discerning power, we would expect that the logic containing the new operators is less expressive than $\mathcal{HL}(\downarrow)$.

2 Syntax and Semantics for Memory Logics

In this section we will introduce the syntax and semantics of the different memory logics that we will discuss in the article, and fix some terminology.

All the languages we will introduce are obtained by extending (in some cases, also slightly modifying) the syntax and semantics of the basic modal logic. Furthermore, with the exception of one case in which we discuss using a stack as a memory container, all the

logics we analyze have the operators ⓡ and ⓚ. Therefore, for notational convention, we will use $\mathcal{ML}$ (for *memory logics*) as a prefix indicating a language that uses a set as a container, and that includes ⓡ and ⓚ. Then we will list the additional operators included in the language. Since the usual semantics of the diamond operator is going to be slightly modified in some cases, we will also include the diamond explicitly in this list. For example, $\mathcal{ML}(\langle r\rangle)$ is basic modal logic (i.e., with the usual diamond operator) extended with ⓡ and ⓚ.

Definition 1 (Syntax)**.** *Let* PROP $= \{p_1, p_2, \dots\}$ *(the* propositional symbols*) and* REL $= \{r_1, r_2, \dots\}$ *(the* relational symbols*) be disjoint, countable infinite sets. The set* FORMS *of formulas in the signature* $\langle$PROP, REL$\rangle$ *is defined as:*

$$\text{FORMS} ::= \top \mid p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \langle r\rangle\varphi \mid \langle\!\langle r\rangle\!\rangle\varphi \mid \text{ⓚ} \mid \text{ⓡ}\varphi$$

where $p \in$ PROP, $r \in$ REL *and* $\varphi, \varphi_1, \varphi_2 \in$ FORMS*. The other standard operators are introduced via definitions. In particular* $[r]\varphi := \neg\langle r\rangle\neg\varphi$ *and* $[\![r]\!]\varphi := \neg\langle\!\langle r\rangle\!\rangle\neg\varphi$*.*

Throughout this article we are going to use the usual notion of *modal depth* of a formula, that is, the deepest nesting of modal operators. Modal formulas without modal operators have a modal depth of zero.

Definition 2 (Semantics)**.** *Given a signature* $\mathcal{S} = \langle$PROP, REL$\rangle$*, a model is a tuple* $\mathcal{M} = \langle D, \mathcal{I}, M\rangle$ *where* D *is a nonempty set,* $\mathcal{I}$ *is an interpretation function such that* $\mathcal{I}(p) \subseteq D$ *for* $p \in$ PROP *and* $\mathcal{I}(r) \subseteq D \times D$ *for* $r \in$ REL*.* $M \subseteq D$ *will be called the* memory *of the model. For notational convenience, let us assume fixed for the rest of the article the models* $\mathcal{M} = \langle D, \mathcal{I}, M\rangle$*,* $\mathcal{M}_1 = \langle D_1, \mathcal{I}_1, M_1\rangle$ *and* $\mathcal{M}_2 = \langle D_2, \mathcal{I}_2, M_2\rangle$*.*

Given a model $\mathcal{M}$ *and a list of states* $[w_1, \dots, w_n]$*,* $w_i \in D$*, we define* $\mathcal{M}[w_1, \dots, w_n] = \langle D, \mathcal{I}, M \cup \{w_1, \dots, w_n\}\rangle$*. Now, let* $\mathcal{M}$ *be a model and* $w \in D$*, then the semantics for the different operators is defined as:*

$$\begin{array}{lll}
\mathcal{M}, w \models \top & \textit{iff} & \textit{always} \\
\mathcal{M}, w \models p & \textit{iff} & w \in \mathcal{I}(p) \\
\mathcal{M}, w \models \neg\varphi & \textit{iff} & \mathcal{M}, w \not\models \varphi \\
\mathcal{M}, w \models \varphi \wedge \psi & \textit{iff} & \mathcal{M}, w \models \varphi \textit{ and } \mathcal{M}, w \models \psi \\
\mathcal{M}, w \models \langle r\rangle\varphi & \textit{iff} & \textit{there is } w' \textit{ such that } (w, w') \in \mathcal{I}(r) \textit{ and } \mathcal{M}, w' \models \varphi \\
\mathcal{M}, w \models \langle\!\langle r\rangle\!\rangle\varphi & \textit{iff} & \textit{there is } w' \textit{ such that } (w, w') \in \mathcal{I}(r) \textit{ and } \mathcal{M}[w], w' \models \varphi \\
\mathcal{M}, w \models \text{ⓡ}\varphi & \textit{iff} & \mathcal{M}[w], w \models \varphi \\
\mathcal{M}, w \models \text{ⓚ} & \textit{iff} & w \in M.
\end{array}$$

Given a model $\mathcal{M}$ *and* $w \in D$*, the set of propositions that are true at a given state* w *is defined as* $\mathsf{props}(w) = \{p \in \text{PROP} \mid w \in \mathcal{I}(p)\}$*. Given two models* $\mathcal{M}_1$ *and* $\mathcal{M}_2$*, and states* $w_1 \in D_1$ *and* $w_2 \in D_2$*, we say that they* agree *when* $\mathsf{props}(w_1) = \mathsf{props}(w_2)$ *and* $w_1 \in M_1$ *iff* $w_2 \in M_2$*.*

Given a model $\mathcal{M}$ *and* w *in the domain of* $\mathcal{M}$*, we call* $\langle \mathcal{M}, w\rangle$ *a* pointed model*.*

A particularly interesting class of models to investigate is the class $\mathcal{C}_\emptyset = \{\mathcal{M} \mid \mathcal{M} = \langle D, \mathcal{I}, \emptyset \rangle\}$, i.e., the class of models where the memory is empty. Since we are working with logics that deal with the notion of state, it is natural to consider starting to evaluate a formula in a model of $\mathcal{C}_\emptyset$. It is over $\mathcal{C}_\emptyset$ that the operators ⓚ and ⓡ have the most natural interpretation, and as we will see in the next sections, the restriction to this class has important effects on expressivity and decidability. It is worth noting that in this case a formula is *initially* evaluated in a model of $\mathcal{C}_\emptyset$, but during the evaluation the model can change to one with nonempty memory. We will put an empty set as a subscript on the prefix $\mathcal{ML}$ every time we work with $\mathcal{C}_\emptyset$ as the class of initial models. For example $\mathcal{ML}(\langle r \rangle)$ restricted to this class of initial models is $\mathcal{ML}_\emptyset(\langle r \rangle)$.

We will not consider all possible combinations of operators, since it is not our intention to be completely exhaustive. We are only going to analyze some combinations that we consider interesting, and in each section we will indicate the fragments we will be using. In many cases, the results shown for some fragments can be easily transferred to other fragments, not explicitly analyzed.

3 Model Equivalence

In this section we will investigate the notion of model equivalence for some of the memory logics that we introduced. Our goal is to define tools that will help us investigate their expressive power. In particular, we will define a notion of model equivalence in terms of Ehrenfeucht-Fraïssé games (Ebbinghaus et al., 1984) and then introduce an alternative, but equivalent, notion in terms of bisimulations.

Definition 3 (Ehrenfeucht-Fraïssé Games)**.** *Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be two models and let $w_1 \in D_1$ and $w_2 \in D_2$.*

An Ehrenfeucht-Fraïssé game *$EF(\mathcal{M}_1, \mathcal{M}_2, w_1, w_2)$ is defined as follows. There are two players called* Spoiler *and* Duplicator*. Duplicator immediately looses the game*

$$EF(\mathcal{M}_1, \mathcal{M}_2, w_1, w_2)$$

if w_1 and w_2 do not agree (i.e., either $\mathsf{props}(w_1) \neq \mathsf{props}(w_2)$ or one of the states is in the memory and the other is not). Otherwise, the game starts, with the players moving alternatively. Spoiler always starts a turn of the game choosing in which model he will make a move. Let us set $s = 1$ and $d = 2$ in case he chooses $\mathcal{M}_1$; otherwise, let $s = 2$ and $d = 1$.

For the logics $\mathcal{ML}(\langle r \rangle)$ and $\mathcal{ML}_\emptyset(\langle r \rangle)$, the possible moves are as follows:

1. Memorize*: Spoiler extends M_s to $M_s \cup \{w_s\}$. The next turn then starts with*

$$EF(\mathcal{M}_1[w_1], \mathcal{M}_2[w_2], w_1, w_2)$$

(Duplicator does nothing in this case).

2. Chose Successor: *Spoiler chooses* $r \in \text{REL}$*, and* v_s*, an* $\mathcal{I}_s(r)$*-successor of* w_s*. If* w_s *has no* $\mathcal{I}_s(r)$*-successors, then Duplicator wins. Duplicator has to chose* v_d*, an* $\mathcal{I}_d(r)$*-successor of* w_d*, such that* v_s *and* v_d *agree. If there is no such successor, Spoiler wins. Otherwise the game continues with* $EF(\mathcal{M}_1, \mathcal{M}_2, v_1, v_2)$*.*

The moves for the logics $\mathcal{ML}(\langle\langle r \rangle\rangle)$ *and* $\mathcal{ML}_\emptyset(\langle\langle r \rangle\rangle)$ *are similar, except that during a* chose successor *step Spoiler always remembers the current world, i.e., the game continues with* $EF(\mathcal{M}_1[w_1], \mathcal{M}_2[w_2], v_1, v_2)$ *after Duplicator response.*

In the case of an infinite game, Duplicator wins. Note that with this definition, exactly one of Spoiler or Duplicator wins each game.

Given two pointed models $\langle \mathcal{M}_1, w_1 \rangle$ *and* $\langle \mathcal{M}_2, w_2 \rangle$ *we write* $\langle \mathcal{M}_1, w_1 \rangle \equiv^{EF} \langle \mathcal{M}_2, w_2 \rangle$ *when Duplicator has a winning strategy for* $EF(\mathcal{M}_1, \mathcal{M}_2, w_1, w_2)$ *(the exact type of game involved will usually be clear from the context, and we will write* $\equiv^{EF}_{\mathcal{L}}$ *when we need to specify that the game corresponds to the language of the logic* $\mathcal{L}$*).*

Even thought in the rest of the article we will use the game notion of model equivalence, a structural notion can be given that is closer to the usual notion of bisimulation for modal logics. Both definitions are equivalent, but depending on the context, one can be more natural than the other (e.g., in Mera (2009) the structural notion is used to prove results related to Craig interpolation).

Definition 4 (Bisimulations)**.** *Let* $\mathcal{M}_1$ *and* $\mathcal{M}_2$ *be two models. Let* $\sim$ *be a binary relation between* $\wp(D_1) \times D_1$ *and* $\wp(D_2) \times D_2$*.*

For $\mathcal{ML}(\langle r \rangle)$ *and* $\mathcal{ML}_\emptyset(\langle r \rangle)$ *a* bisimulation *satisfies the following properties:*

(*nontriv*) $\sim$ *is not empty.*

(*agree*) *If* $\langle M, m \rangle \sim \langle N, n \rangle$*, then* m *and* n *agree.*

(*forth*) *If* $\langle M, m \rangle \sim \langle N, n \rangle$ *and* $(m, m') \in \mathcal{I}^1(r)$*, then there exists* $n' \in D_2$ *such that* $(n, n') \in \mathcal{I}^2(r)$ *and* $\langle M, m' \rangle \sim \langle N, n' \rangle$*.*

(*back*) *If* $\langle M, m \rangle \sim \langle N, n \rangle$ *and* $(n, n') \in \mathcal{I}^2(r)$*, then there exists* $m' \in D_1$ *such that* $(m, m') \in \mathcal{I}^1(r)$ *and* $\langle M, m' \rangle \sim \langle N, n' \rangle$*.*

(*remember*) *If* $\langle M, m \rangle \sim \langle N, n \rangle$*, then* $\langle M \cup \{m\}, m \rangle \sim \langle N \cup \{n\}, n \rangle$*.*

For the logics $\mathcal{ML}(\langle\langle r \rangle\rangle)$ *and* $\mathcal{ML}_\emptyset(\langle\langle r \rangle\rangle)$ *the* (back) *and* (forth) *conditions are replaced by:*

(*mforth*) *If* $\langle M, m \rangle \sim \langle N, n \rangle$ *and* $(m, m') \in \mathcal{I}^1(r)$*, then there exists* $n' \in D_2$ *such that* $(n, n') \in \mathcal{I}^2(r)$ *and* $\langle M \cup \{m\}, m' \rangle \sim \langle N \cup \{n\}, n' \rangle$*.*

(*mback*) *If* $\langle M, m \rangle \sim \langle N, n \rangle$ *and* $(n, n') \in \mathcal{I}^2(r)$*, then there exists* $m' \in D_1$ *such that* $(m, m') \in \mathcal{I}^1(r)$ *and* $\langle M \cup \{m\}, m' \rangle \sim \langle N \cup \{n\}, n' \rangle$*.*

Given two pointed models $\langle \mathcal{M}_1, w_1\rangle$ and $\langle \mathcal{M}_2, w_2\rangle$ we write $\langle \mathcal{M}_1, w_1\rangle \leftrightarrows \langle \mathcal{M}_2,\ w_2\rangle$ if there is a bisimulation linking $\langle M_1, w_1\rangle$ and $\langle M_2, w_2\rangle$. Again, the exact type of bisimulation involved will usually be clear from the context, and we will write $\leftrightarrows_{\mathcal{L}}$ when we need to specify that the bisimulation corresponds to the logic $\mathcal{L}$.

As we said before, the notions of Ehrenfeucht-Fraïssé games and bisimulations coincide, as indicated in the following theorem.

Theorem 5. *Let $\mathcal{L} \in \{\mathcal{ML}(\langle r\rangle), \mathcal{ML}_\emptyset(\langle r\rangle), \mathcal{ML}(\langle\langle r\rangle\rangle), \mathcal{ML}_\emptyset(\langle\langle r\rangle\rangle)\}$. Given two pointed models $\langle \mathcal{M}_1, w_1\rangle$ and $\langle \mathcal{M}_2, w_2\rangle$ then $\langle \mathcal{M}_1,\ w_1\rangle \equiv^{EF}_{\mathcal{L}} \langle \mathcal{M}_2, w_2\rangle$ if and only if $\langle \mathcal{M}_1, w_1\rangle \leftrightarrows_{\mathcal{L}} \langle \mathcal{M}_2, w_2\rangle$.*

Proof. We will discuss the case only for $\mathcal{ML}(\langle r\rangle)$ as the proof is similar for languages containing $\langle\langle r\rangle\rangle$.

For the right to left direction. Assume that $\langle \mathcal{M}_1, w_1\rangle \leftrightarrows \langle \mathcal{M}_2, w_2\rangle$ and that $\sim$ is a bisimulation linking $\langle M_1, w_1\rangle$ and $\langle M_2, w_2\rangle$. We will prove that there is a strategy for Duplicator in the game $EF(\mathcal{M}_1, \mathcal{M}_2, w_1, w_2)$. First note that the game $EF(\mathcal{M}_1, \mathcal{M}_2, w_1, w_2)$ is well defined, since by *(agree)*, w_1 and w_2 are agreeing states. We show that there is a strategy for Duplicator by proving that (1) for any pair of tuples $\langle S, w\rangle$ and $\langle Q, v\rangle$ such that $\langle S, w\rangle \sim \langle Q, v\rangle$, and for any move Spoiler makes in the game $EF(\mathcal{M}_1[S], \mathcal{M}_2[Q], w, v)$, there is always an appropriate answer for Duplicator such that the next step of the game is $EF(\mathcal{M}_1[S'], \mathcal{M}_2[Q'], w', v')$ and $\langle S', w'\rangle \sim \langle Q', v'\rangle$. Given the initial assumptions, the fact that Duplicator has a winning strategy on the game $EF(\mathcal{M}_1, \mathcal{M}_2, w_1, w_2)$ easily follows from (1). So let us suppose that $\langle S, w\rangle \sim \langle Q, v\rangle$ and consider the game $EF(\mathcal{M}_1[S], \mathcal{M}_2[Q], w, v)$. Without loss of generality, we assume that Spoiler chooses $\mathcal{M}_1$ to make his move. There are two kinds of moves Spoiler can do:

- Spoiler make a memorize step, and the game continues with $EF(\mathcal{M}_1[S\cup\{w\}], \mathcal{M}_2[Q\cup\{v\}], w, v)$. By the *(remember)* condition, we know that $\langle S\cup\{w\}, w\rangle \sim \langle Q\cup\{v\}, v\rangle$.
- Spoiler chooses an r-successor w' of w. By the *(forth)* condition (we use *(back)* here if Duplicator chooses $\mathcal{M}_2$ for his move), there is an r-successor v' of v such that $\langle S, w'\rangle \sim \langle Q, v'\rangle$. Using *(agree)*, we know that w' and v' agree, so v' is a good choice for Duplicator. The game continues with $EF(\mathcal{M}_1[S], \mathcal{M}_2[Q], w', v')$ and $\langle S, w'\rangle \sim \langle Q, v'\rangle$.

For the other direction, suppose that Duplicator has a winning strategy $\mathcal{S}$ on the game $EF(\mathcal{M}_1, \mathcal{M}_2, w_1, w_2)$. We define $\sim$ in the following way: $\langle S, w\rangle \sim \langle Q, v\rangle$ if and only if $EF(\mathcal{M}_1[S], \mathcal{M}_2[Q], w, v)$ is a reachable state of $EF(\mathcal{M}_1, \mathcal{M}_2, w_1,\ w_2)$ when Duplicator follows strategy $\mathcal{S}$. We have to prove that the relation $\sim$ is a bisimulation. Suppose that $\langle S, w\rangle \sim \langle Q, v\rangle$.

- The condition *(agree)* is easy to check.
- To see that the *(forth)* condition holds, suppose that $(m, m') \in \mathcal{I}_1(r)$. One possible move for Spoiler in the game $EF(\mathcal{M}_1[S], \mathcal{M}_2[Q], w, v)$ is to choose m' from $\mathcal{M}_1$, and

because Duplicator uses the winning strategy $\mathcal{S}$, he can answer with a state $v' \in \mathcal{M}_2$, a successor of v, such that w' and v' agree. Therefore, the next step of the game is $EF(\mathcal{M}_1[S], \mathcal{M}_2[Q], w', v')$, and by definition, $\langle S, w' \rangle \sim \langle Q, v' \rangle$. The *(back)* condition is equivalent.

- Finally, to verify the *(remember)* condition, note that in the game $EF(\mathcal{M}_1[S], \mathcal{M}_2[Q], w, v)$ Spoiler can choose to make a memorize step, and therefore the next step of the game is $E(\mathcal{M}_1[S \cup \{w\}], \mathcal{M}_2[Q \cup \{v\}], w, v)$. By definition, that means that $\langle S \cup \{w\}, w \rangle \sim \langle Q \cup \{v\}, v \rangle$.

Therefore, $\sim$ is actually a bisimulation. Because the state $EF(\mathcal{M}_1, \mathcal{M}_2, w_1, w_2)$ is (trivially) reachable, $\langle M_1, w_1 \rangle \sim \langle M_2, w_2 \rangle$ as desired. □

As one could expect, both notions of model equivalence preserve the truth value of formulas. Given two pointed models $\langle \mathcal{M}_1, w_1 \rangle$ and $\langle \mathcal{M}_2, w_2 \rangle$, we write $\langle \mathcal{M}_1, w_1 \rangle \equiv_{\mathcal{L}} \langle \mathcal{M}_2, w_2 \rangle$ if for any formula φ in the language of the logic $\mathcal{L}$ we have that $\mathcal{M}_1, w_1 \models \varphi$ if and only if $\mathcal{M}_2, w_2 \models \varphi$. Proving then that $\langle \mathcal{M}_1, w_1 \rangle \equiv^{EF}_{\mathcal{L}} \langle \mathcal{M}_2, w_2 \rangle$ (equivalently $\langle \mathcal{M}_1, w_1 \rangle \leftrightarrows_{\mathcal{L}} \langle \mathcal{M}_2, w_2 \rangle$) implies $\langle \mathcal{M}_1, w_1 \rangle \equiv_{\mathcal{L}} \langle \mathcal{M}_2, w_2 \rangle$ only requires a simple induction. Establishing that the notions $\equiv^{EF}_{\mathcal{L}}$, $\leftrightarrows_{\mathcal{L}}$ and $\equiv_{\mathcal{L}}$ coincide on image finite models (i.e., models where each state has only a finite number of successors considering the union of the accessibility relations) is only slightly harder.

Theorem 6. *Let $\mathcal{L} \in \{\mathcal{ML}(\langle r \rangle), \mathcal{ML}_\emptyset(\langle r \rangle), \mathcal{ML}(\langle\!\langle r \rangle\!\rangle), \mathcal{ML}_\emptyset(\langle\!\langle r \rangle\!\rangle)\}$. Let $\langle \mathcal{M}_1, w_1 \rangle$ and $\langle \mathcal{M}_2, w_2 \rangle$ be two pointed models. Then $\langle \mathcal{M}_1, w_1 \rangle \equiv^{EF}_{\mathcal{L}} \langle \mathcal{M}_2, w_2 \rangle$ (equivalently, $\langle \mathcal{M}_1, w_1 \rangle \leftrightarrows_{\mathcal{L}} \langle \mathcal{M}_2, w_2 \rangle$) implies $\langle \mathcal{M}_1, w_1 \rangle \equiv_{\mathcal{L}} \langle \mathcal{M}_2, w_2 \rangle$. If $\mathcal{M}_1$ and $\mathcal{M}_2$ are image finite, then $\langle \mathcal{M}_1, w_1 \rangle \equiv_{\mathcal{L}} \langle \mathcal{M}_2, w_2 \rangle$ implies both $\langle \mathcal{M}_1, w_1 \rangle \equiv^{EF}_{\mathcal{L}} \langle \mathcal{M}_2, w_2 \rangle$ and $\langle \mathcal{M}_1, w_1 \rangle \leftrightarrows_{\mathcal{L}} \langle \mathcal{M}_2, w_2 \rangle$.*

4 Expressive Power

In this section we compare the expressive power of memory logics with respect to both modal and hybrid logics. To do this, we will have to find a natural mapping between models of each logic. Such a mapping is easy to define in the case of the $\mathcal{ML}_\emptyset$ logics, where we only consider models with an empty memory: each modal model $\langle D, \mathcal{I} \rangle$ can be identified with the memory model $\langle D, \mathcal{I}, \emptyset \rangle$. Similarly, for sentences of $\mathcal{HL}(\downarrow)$ (i.e., formulas where each nominal i appears in the scope of $\downarrow i$) the memory model $\langle D, \mathcal{I}, \emptyset \rangle$ can be identified with the hybrid model $\langle D, \mathcal{I}, g \rangle$ for g an arbitrary assignment. In other cases, the definition will involve a change in the signature. But for the moment, assume that we consider two logics $\mathcal{L}$ and $\mathcal{L}'$ such that both can be evaluated over the same class of models (modulo representation issues).

Definition 7 ($\mathcal{L} \leq \mathcal{L}'$)**.** *We say that $\mathcal{L}'$ is* at least as expressive as *$\mathcal{L}$ (notation $\mathcal{L} \leq \mathcal{L}'$) if there is a function Tr between formulas of $\mathcal{L}$ and $\mathcal{L}'$ such that for every model $\mathcal{M}$ and every formula φ of $\mathcal{L}$ we have that*

$$\mathcal{M} \models_{\mathcal{L}} \varphi \text{ iff } \mathcal{M} \models_{\mathcal{L}'} \mathsf{Tr}(\varphi),$$

(here it should be understood that the model $\mathcal{M}$ is seen as a model of $\mathcal{L}$ on the left and as a model of $\mathcal{L}'$ on the right, and that we use in each case the appropriate semantic relation $\models_{\mathcal{L}}$ or $\models_{\mathcal{L}'}$ as required).

We say that $\mathcal{L}'$ is strictly more expressive *than $\mathcal{L}$ (notation $\mathcal{L} < \mathcal{L}'$) if $\mathcal{L} \leq \mathcal{L}'$ but not $\mathcal{L}' \leq \mathcal{L}$. And we say that $\mathcal{L}$ and $\mathcal{L}'$ are equally expressive (notation $\mathcal{L} = \mathcal{L}'$) if $\mathcal{L} \leq \mathcal{L}'$ and $\mathcal{L}' \leq \mathcal{L}$.*

To improve the presentation of this section, sometimes we are going to present theorems that are later subsumed by stronger results (e.g. Theorem 10 is subsumed by Theorem 13, and later by Corollary 21). The reasons for doing this are in some cases just for the sake of clarity. In others it is because we believe that the proofs of some results are interesting by themselves.

4.1 Logics with an initially empty memory

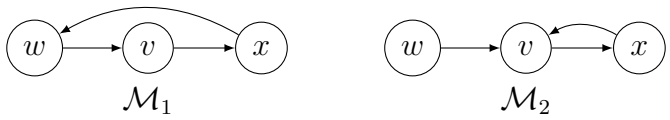
We will compare the logics $\mathcal{ML}_{\emptyset}$ with the basic modal logic $\mathcal{K}$ and the hybrid logic $\mathcal{HL}(\downarrow)$. We are going to establish that $\mathcal{K} < \mathcal{ML}_{\emptyset}(\langle\!\langle r \rangle\!\rangle) < \mathcal{ML}_{\emptyset}(\langle r \rangle) < \mathcal{HL}(\downarrow)$.

First we are going to show that the freedom to decide when to remember a state gives $\mathcal{ML}_{\emptyset}(\langle r \rangle)$ more expressive power when compared to $\mathcal{ML}_{\emptyset}(\langle\!\langle r \rangle\!\rangle)$.

Theorem 8. $\mathcal{ML}_{\emptyset}(\langle\!\langle r \rangle\!\rangle) < \mathcal{ML}_{\emptyset}(\langle r \rangle)$.

Proof. $[\mathcal{ML}_{\emptyset}(\langle\!\langle r \rangle\!\rangle) \leq \mathcal{ML}_{\emptyset}(\langle r \rangle)]$: It is easy to see that there is a translation Tr from $\mathcal{ML}_{\emptyset}(\langle\!\langle r \rangle\!\rangle)$ to $\mathcal{ML}_{\emptyset}(\langle r \rangle)$-formulas which maps $\langle\!\langle r \rangle\!\rangle\varphi$ to $\textcircled{\scriptsize r}\langle r \rangle\varphi$ and verifies $\mathcal{M} \models \varphi$ if and only if $\mathcal{M} \models \mathsf{Tr}(\varphi)$.

$[\mathcal{ML}_{\emptyset}(\langle r \rangle) \not\leq \mathcal{ML}_{\emptyset}(\langle\!\langle r \rangle\!\rangle)]$: Let $\mathcal{M}_1 = \langle \{w, v, x\}, \mathcal{I}_1, \emptyset \rangle$ and $\mathcal{M}_2 = \langle \{w, v, x\}, \mathcal{I}_2, \emptyset \rangle$ such that $\mathcal{I}_1(r) = \{(w, v), (v, x), (x, w)\}$, $\mathcal{I}_2(r) = \{(w, v), (v, x), (x, v)\}$, and $\mathcal{I}_1(p) = \mathcal{I}_2(p) = \emptyset$ for $p \in \text{PROP}$ as shown below:



We claim $\langle \mathcal{M}_1, w \rangle \equiv^{EF}_{\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)} \langle \mathcal{M}_2, w \rangle$. As every state in both models has a unique successor, Duplicator has only one way of playing, which is actually a winning strategy. Hence $\langle \mathcal{M}_1, w \rangle \equiv_{\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)} \langle \mathcal{M}_2, w \rangle$. But $\mathcal{M}_1, w \not\models \langle r \rangle\textcircled{\scriptsize r}\langle r \rangle\langle r \rangle\textcircled{\scriptsize k}$, while $\mathcal{M}_2, w \models \langle r \rangle\textcircled{\scriptsize r}\langle r \rangle\langle r \rangle\textcircled{\scriptsize k}$. □

We will now compare the expressive power of memory logics with the basic modal logic $\mathcal{K}$. It is not difficult to see intuitively that $\textcircled{\scriptsize r}$ and $\textcircled{\scriptsize k}$ do bring additional expressive power into the language of $\mathcal{K}$: with their help we can detect cycles in a given model, while formulas of $\mathcal{K}$ are invariant under unraveling.

Theorem 9. $\mathcal{K} < \mathcal{ML}_{\emptyset}(\langle\!\langle r \rangle\!\rangle)$.

Proof. As $\mathcal{K}$ is a sub-language of $\mathcal{ML}_\emptyset(\langle\!\langle r \rangle\!\rangle)$, $\mathcal{K} \leq \mathcal{ML}_\emptyset(\langle\!\langle r \rangle\!\rangle)$ taking Tr to be the identity function. To see that $\mathcal{ML}_\emptyset(\langle\!\langle r \rangle\!\rangle) \not\leq \mathcal{K}$, let $\mathcal{M}_1 = \langle \{w\}, \mathcal{I}_1, \emptyset \rangle$ with $\mathcal{I}_1(r) = \{(w,w)\}$, $\mathcal{M}_2 = \langle \{u,v\}, \mathcal{I}_2, \emptyset \rangle$ with $\mathcal{I}_2 = \{(u,v),(v,u)\}$, and $\mathcal{I}_1(p) = \mathcal{I}_2(p) = \emptyset$ for $p \in \mathsf{props}$ be two models as shown below:

$\mathcal{M}_1$ $\mathcal{M}_2$

The models are $\mathcal{K}$ bisimilar (Blackburn et al., 2001). However, they can be distinguished by the $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$-formula $\langle\!\langle r \rangle\!\rangle \textcircled{k}$. □

We will now compare the expressive power of memory logics with respect to hybrid logics. The most natural choice for the comparison is the hybrid logic $\mathcal{HL}(\downarrow)$. We will prove that $\mathcal{HL}(\downarrow)$ is strictly more expressive than $\mathcal{ML}_\emptyset(\langle r \rangle)$. Intuitively, $\downarrow$ can easily simulate $\textcircled{r}$, but $\textcircled{k}$ does not distinguish between different memorized states (while nominals bound by $\downarrow$ do).

Theorem 10. $\mathcal{ML}_\emptyset(\langle r \rangle) < \mathcal{HL}(\downarrow)$.

Proof. We first prove that $\mathcal{ML}_\emptyset(\langle r \rangle) \leq \mathcal{HL}(\downarrow)$. We define the translation Tr, taking $\mathcal{ML}_\emptyset(\langle r \rangle)$-formulas over the signature $\langle \textsc{prop}, \textsc{rel} \rangle$ to $\mathcal{HL}(\downarrow)$ sentences over the signature $\langle \textsc{prop}, \textsc{rel}, \textsc{nom} \rangle$. Tr is defined for any finite set $N \subseteq \textsc{nom}$ as follows:

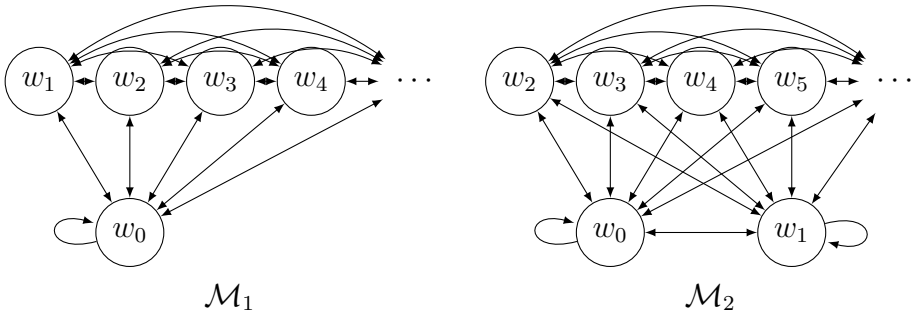
$$\begin{array}{rcl}
\mathsf{Tr}_N(p) & = & p \quad p \in \textsc{prop} \\
\mathsf{Tr}_N(\textcircled{k}) & = & \bigvee_{i \in N} i \\
\mathsf{Tr}_N(\neg\varphi) & = & \neg \mathsf{Tr}_N(\varphi) \\
\mathsf{Tr}_N(\varphi_1 \wedge \varphi_2) & = & \mathsf{Tr}_N(\varphi_1) \wedge \mathsf{Tr}_N(\varphi_2) \\
\mathsf{Tr}_N(\langle r \rangle \varphi) & = & \langle r \rangle \mathsf{Tr}_N(\varphi) \\
\mathsf{Tr}_N(\textcircled{r}\varphi) & = & \downarrow i.\mathsf{Tr}_{N \cup \{i\}}(\varphi) \quad \text{where } i \notin N.
\end{array}$$

Induction then shows that $\mathcal{M}, w \models \varphi$ iff $\mathcal{M}, g, w \models \mathsf{Tr}_\emptyset(\varphi)$, for any g.

Now we prove that $\mathcal{HL}(\downarrow)$ is strictly more expressive than $\mathcal{ML}_\emptyset(\langle r \rangle)$. Let

$$\begin{array}{rcl}
\mathcal{M}_1 & = & \langle \{w_0, w_1, w_2, \dots\}, \mathcal{I}_1, \emptyset \rangle \\
\mathcal{M}_2 & = & \langle \{w_0, w_1, w_2, \dots\}, \mathcal{I}_2, \emptyset \rangle \\
\mathcal{I}_1(r) & = & \{(n,m) \mid n \neq m\} \cup \{(w_0, w_0)\} \\
\mathcal{I}_2(r) & = & \mathcal{I}_1(r) \cup \{(w_1, w_1)\} \\
\mathcal{I}_1(p) = \mathcal{I}_2(p) & = & \emptyset \quad \text{for } p \in \textsc{prop}.
\end{array}$$

Graphically,



We prove that $\langle \mathcal{M}_1, w_0 \rangle \equiv^{EF}_{\mathcal{ML}_\emptyset(\langle r \rangle)} \langle \mathcal{M}_2, w_0 \rangle$ showing a winning strategy for Duplicator. Intuitively, the strategy is as follows: whenever one player is in $\langle \mathcal{M}_1, w_0 \rangle$ the other will be in $\langle \mathcal{M}_2, w_0 \rangle$ or $\langle \mathcal{M}_2, w_1 \rangle$, and conversely whenever a player is in $\langle \mathcal{M}_1, w_n \rangle$, $n > 0$, the other will be in $\langle \mathcal{M}_2, w_m \rangle$, $m > 1$. This is maintained until Spoiler (if ever) decides to remember a state. Once this is done, then any move leads to a win of Duplicator. Formally, the winning strategy will have two stages:

1. While Spoiler does not remember any reflexive state, Duplicator plays as follows: if Spoiler chooses w_0 in any model, Duplicator chooses w_0 in the other; if Spoiler chooses $w_n, n > 0$ in $\mathcal{M}_1$, Duplicator plays w_{n+1} in $\mathcal{M}_2$; if Spoiler chooses $w_n, n > 0$ in $\mathcal{M}_2$, Duplicator plays w_{n-1} in $\mathcal{M}_1$. Notice that with this strategy Spoiler chooses a reflexive state if and only if Duplicator answers with a reflexive one. This is clearly a winning strategy.

2. If ever Spoiler decides to remember a reflexive state, Duplicator starts using the following strategy: if Spoiler selects a state w_n, Duplicator answers with an agreeing state w_m of the opposite model. Notice that this is always possible since both w_n and w_m see infinitely many non remembered states and at least one remembered state.

On the other hand, let φ be the formula $\downarrow i.\langle r \rangle (i \wedge \langle r \rangle (\neg i \wedge \downarrow i.\langle r \rangle i))$. It is easy to see that $\mathcal{M}_1, w_0 \not\models \varphi$ but $\mathcal{M}_2, w_0 \models \varphi$. □

We have shown that $\mathcal{ML}_\emptyset(\langle r \rangle) < \mathcal{HL}(\downarrow)$ but the proof seems to intrinsically use infinite models, in contrast with the proofs for Theorems 8, 9 and 10 in which finite models are used. Actually, $\mathcal{ML}_\emptyset(\langle r \rangle) < \mathcal{HL}(\downarrow)$ even on *finite models*. For this purpose we will first introduce a version of the Ehrenfeucht-Fraïssé game presented in Definition 3 where the number of turns is bounded.

Definition 11. *The* n-moves Ehrenfeucht-Fraïssé game *for a given logic* $\mathcal{L}$*, denoted*

$$EF^n_{\mathcal{L}}(\mathcal{M}_1, \mathcal{M}_2, w_1, w_2),$$

is the game in which Spoiler can only make n *moves in the game to beat Duplicator. If Duplicator has a strategy to remain undefeated for* n *moves, he wins the game and we write* $\langle \mathcal{M}_1, w_1 \rangle \equiv^{EF^n}_{\mathcal{L}} \langle \mathcal{M}_2, w_2 \rangle$.

We will state without a proof the following easy theorem.

Theorem 12. *Let* $\mathcal{L} \in \{\mathcal{ML}(\langle r\rangle), \mathcal{ML}_\emptyset(\langle r\rangle), \mathcal{ML}(\langle\!\langle r\rangle\!\rangle), \mathcal{ML}_\emptyset(\langle\!\langle r\rangle\!\rangle)\}$. *For any pair of pointed models,* $\langle \mathcal{M}_1, w_1\rangle \equiv_{\mathcal{L}}^{EF^n} \langle \mathcal{M}_2, w_2\rangle$ *if and only if for every formula* φ *of* $\mathcal{L}$ *with modal depth* n, $\mathcal{M}_1, w_1 \models \varphi$ *iff* $\mathcal{M}_2, w_2 \models \varphi$.

Now we can prove the desired result for finite models:

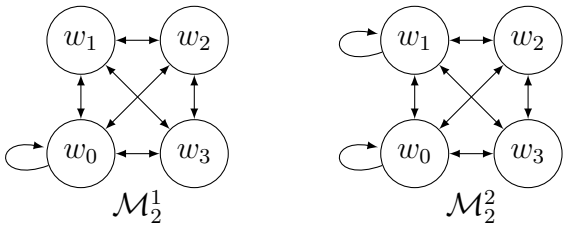
Theorem 13. $\mathcal{ML}_\emptyset(\langle r\rangle) < \mathcal{HL}(\downarrow)$ *over the class of finite models.*

Proof. We will prove that there is a property φ expressible in $\mathcal{HL}(\downarrow)$ that cannot be expressed in $\mathcal{ML}_\emptyset(\langle r\rangle)$ over finite models. To do this, for every n we will exhibit two finite models $\mathcal{M}_1^n, \mathcal{M}_2^n$ such that $\mathcal{M}_1^n, w_0 \models \varphi$, $\mathcal{M}_2^n, w_0 \not\models \varphi$ but $\langle \mathcal{M}_1^n, w_0\rangle \equiv_{\mathcal{ML}_\emptyset(\langle r\rangle)}^{EF^n} \langle \mathcal{M}_2^n, w_0\rangle$. This implies that there is no $\mathcal{ML}_\emptyset(\langle r\rangle)$-formula ψ capable of expressing this property.

Let $\varphi = \downarrow i.\langle r\rangle(i \wedge \langle r\rangle(\neg i \wedge \downarrow i.\langle r\rangle i))$ as in the proof of Theorem 10, and let, for $n \geq 1$, $\mathcal{M}_1^n = \langle D^n, \mathcal{I}_1^n, \emptyset\rangle$ and $\mathcal{M}_2^n = \langle D^n, \mathcal{I}_2^n, \emptyset\rangle$ where

$$\begin{aligned} D^n &= \{w_0, \ldots, w_{n+1}\}, \\ \mathcal{I}_1^n(r) &= \{(a,b) \mid a,b \in D^n, a \neq b\} \cup \{(w_0, w_0)\}, \\ \mathcal{I}_2^n(r) &= \mathcal{I}_1^n(r) \cup \{(w_1, w_1)\}, \text{ and} \\ \mathcal{I}_1^n(p) = \mathcal{I}_2^n(p) &= \emptyset \text{ for } p \in \text{PROP} \end{aligned}$$

As an example, $\mathcal{M}_1^2$ and $\mathcal{M}_2^2$ would be



Clearly, for every $n \geq 1$, $\mathcal{M}_1^n, w_0 \not\models \varphi$ and $\mathcal{M}_2^n, w_0 \models \varphi$. To prove that $\langle \mathcal{M}_1^n, w_0\rangle \equiv_{\mathcal{ML}_\emptyset(\langle r\rangle)}^{EF^n} \langle \mathcal{M}_2^n, w_0\rangle$, we will describe Duplicator's winning strategy:

1. While Spoiler does not remember any reflexive state, Duplicator plays with the following strategy: whenever Spoiler is in w_k, $2 \leq k \leq n+1$ in one model, Duplicator is in an agreeing state $w_{k'}$, $2 \leq k' \leq n+1$ in the other one. If one player is in w_0 in $\mathcal{M}_1^n$ then the other is in w_0 or w_1 in $\mathcal{M}_2^n$. Finally, if Spoiler plays w_1 in $\mathcal{M}_1^n$, Duplicator plays in an agreeing w_k, $2 \leq k \leq n+1$ in $\mathcal{M}_2^n$. With this strategy, Spoiler chooses a reflexive state if and only if Duplicator answers with a reflexive one, and Duplicator is always able to choose an agreeing state.

2. If ever Spoiler decides to remember a reflexive state, then for every state w_i chosen by Spoiler, Duplicator will always have an agreeing state w_j in the other model. This happens because the models have $n+2$ states, and therefore there is always at least two non-remembered states. At each round the number of unremembered states can only be decremented by one, and then up to round n both players will always see remembered and unremembered states from w_i and well as from w_j.

Because Duplicator wins the game for any n, any candidate $\psi \in \mathcal{ML}_\emptyset(\langle r \rangle)$ expressing φ will fail for a sufficiently large n. □

The $\mathcal{HL}(\downarrow)$-sentence we use in the proofs of Theorem 10 and 13 has only one nominal. Hence, we have actually proved that $\mathcal{HL}_1(\downarrow) \not\leq \mathcal{ML}_\emptyset(\langle r \rangle)$, where $\mathcal{HL}_1(\downarrow)$ is $\mathcal{HL}(\downarrow)$ restricted to only one nominal. But actually, it is also the case that $\mathcal{ML}_\emptyset(\langle r \rangle) \not\leq \mathcal{HL}_1(\downarrow)$. More generally, for any fixed number k of nominals, the logics $\mathcal{HL}_k(\downarrow)$ and $\mathcal{ML}_\emptyset(\langle r \rangle)$ are incomparable.

Theorem 14. *For any fixed k, the logics $\mathcal{HL}_k(\downarrow)$ and $\mathcal{ML}_\emptyset(\langle r \rangle)$ are incomparable in terms of expressive power.*

Proof. We will show the proof for $k = 1$, the general case being similar. $\mathcal{HL}_1(\downarrow) \not\leq \mathcal{ML}_\emptyset(\langle r \rangle)$ is a direct consequence of the proof of Theorem 10.

To prove $\mathcal{ML}_\emptyset(\langle r \rangle) \not\leq \mathcal{HL}_1(\downarrow)$, let $\mathcal{M}_1 = \langle \{w_1, w_2, w_3\}, \mathcal{I}_1, \emptyset \rangle$ with $\mathcal{I}_1(r) = \{(w_i, w_j) \mid 1 \leq i, j \leq 3\}$ and $\mathcal{I}_1(p) = \emptyset$ for $p \in$ PROP, and $\mathcal{M}_2 = \langle \{w_1, w_2\}, \mathcal{I}_2, \emptyset \rangle$ with $\mathcal{I}_2(r) = \{(w_i, w_j) \mid 1 \leq i, j \leq 2\}$ $\mathcal{I}_2(p) = \emptyset$ for $p \in$ PROP. That is, $\mathcal{M}_1$ is a clique of size 3 while $\mathcal{M}_2$ is a clique of size 2. It is easy to check that $\langle \mathcal{M}_1, w_1 \rangle \equiv_{\mathcal{HL}_1(\downarrow)} \langle \mathcal{M}_2, w_1 \rangle$ because they are $\mathcal{HL}_1(\downarrow)$-bisimilar as defined in (Areces & ten Cate, 2006). However, the formula $\varphi = \textcircled{r}\langle r \rangle(\neg \textcircled{k} \wedge \textcircled{r}\langle r \rangle \neg \textcircled{k})$ distinguishes the models: $\mathcal{M}_1, w_1 \models \varphi$ but $\mathcal{M}_2, w_1 \not\models \varphi$.

The proof for $\mathcal{HL}_k(\downarrow)$ is similar, taking cliques of the appropriate size. □

4.2 Erase and forget

As it is natural to define operators that store states in the memory, we can also introduce operators that *delete* states from it. In this section we will investigate their behavior. We extend the memory logics we have been discussing with two new operators that remove states from the memory. We define both a global operator Ⓔ that completely wipes out the memory, and a local version Ⓕ, which deletes the current evaluation state.

We extend the syntax of the memory languages to include Ⓔ and Ⓕ:

$$\text{FORMS} ::= \top \mid p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \langle r \rangle \varphi \mid \langle\langle r \rangle\rangle \varphi \mid \textcircled{k} \mid \textcircled{r}\varphi \mid \textcircled{e}\varphi \mid \textcircled{f}\varphi,$$

were $\varphi \in$ FORMS (see Definition 1 for details). We also extend the semantics (Definition 2) with the following two conditions:

$$\begin{array}{lll} \langle D, \mathcal{I}, M \rangle, w \models \textcircled{e}\varphi & \text{iff} & \langle D, \mathcal{I}, \emptyset \rangle, w \models \varphi \\ \langle D, \mathcal{I}, M \rangle, w \models \textcircled{f}\varphi & \text{iff} & \langle D, \mathcal{I}, M \setminus \{w\} \rangle, w \models \varphi. \end{array}$$

As we intuitively discussed above, the Ⓔ operator replaces the current memory with the empty set, while Ⓕ only removes the current state.

In this section we are only going to consider Ⓔ and Ⓕ for classes of models where the original memory is empty, and with the usual interpretation for the diamond operator. Hence, following our naming convention, we will refer to these logics adding the new

operators to the prefix $\mathcal{ML}$. For example, $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e}, \textcircled{f})$ is the memory logic augmented with both $\textcircled{e}$ and $\textcircled{f}$ operators.

Clearly, the notions of Ehrenfeucht-Fraïssé game and bisimulation need to be extended to include these new operators, given that we want model equivalence to preserve the truth value of formulas.

Definition 15. *The definition of Ehrenfeucht-Fraïssé game for logics with* $\textcircled{e}$ *and* $\textcircled{f}$ *extends Definition 3 adding two new possible moves. Remember that the current move is* $EF(\mathcal{M}_1, \mathcal{M}_2, w_1, w_2)$*, that the turn starts by Spoiler choosing one of the two models, and that we set* $s = 1$ *and* $d = 2$ *in case Spoiler chooses* $\mathcal{M}_1$*, and that* $s = 2$ *and* $d = 1$ *otherwise.*

1. Erase*: Spoiler wipes out the memory, setting* $M_s = M_d = \emptyset$*. The next turn starts with* $EF(\langle D_1, \mathcal{I}_1, \emptyset\rangle, \langle D_2, \mathcal{I}_2, \emptyset\rangle, w_1, w_2)$*.*
2. Forget*: Spoiler deletes* w_s *from* M_s *setting* $M_s = M_s \setminus \{w_s\}$*. The next turn starts with* $EF(\langle D_1, \mathcal{I}_1, M_1 \setminus \{w_1\}\rangle, \langle D_2, \mathcal{I}_2, M_2 \setminus \{w_2\}\rangle, w_1, w_2)$*.*

In a similar way we can extend the notion of bisimulation we introduced before.

Definition 16. *The notion of bisimulation extend the one described in Definition 4 with the rules:*

(*erase*) *If* $\langle M, m\rangle \sim \langle N, n\rangle$*, then* $\langle \emptyset, m\rangle \sim \langle \emptyset, n\rangle$*.*

(*forget*) *If* $\langle M, m\rangle \sim \langle N, n\rangle$*, then* $\langle M \setminus \{m\}, m\rangle \sim \langle N \setminus \{n\}, n\rangle$*.*

Once more, the definitions are modular, each new type of move in the Ehrenfeucht-Fraïssé, and each new rule for the bisimulation definition corresponds, respectively to the $\textcircled{e}$ and $\textcircled{f}$ operators. If one of these operators is added to the language, the corresponding rule or type of move needs to be added to the corresponding definition of bisimulation or game in order to preserve the extended language.

Now we can establish the first result regarding these new operators: independently adding $\textcircled{e}$ and $\textcircled{f}$ does increase the expressive power.

Theorem 17. $\mathcal{ML}_\emptyset(\langle r\rangle) < \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{f})$ *and* $\mathcal{ML}_\emptyset(\langle r\rangle) < \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e})$*.*

Proof. It is trivial to see that both $\mathcal{ML}_\emptyset(\langle r\rangle) \leq \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{f})$ and $\mathcal{ML}_\emptyset(\langle r\rangle) \leq \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e})$ hold using the identity translation. To verify $\mathcal{ML}_\emptyset(\langle r\rangle) \neq \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{f})$ and $\mathcal{ML}_\emptyset(\langle r\rangle) \neq \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e})$, let $\mathcal{M}_1$ and $\mathcal{M}_2$ be the models described in the proof of Theorem 10 Recall that $\langle \mathcal{M}_1, 0\rangle$ is $\mathcal{ML}_\emptyset(\langle r\rangle)$-bisimilar to $\langle \mathcal{M}_2, 0\rangle$.

To see that $\mathcal{ML}_\emptyset(\langle r\rangle) \neq \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{f})$, we show that these two pointed models are distinguishable with a $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{f})$-formula. Let

$$\psi = [r]\textcircled{f}(\langle r\rangle(\textcircled{k} \wedge \langle r\rangle\textcircled{k})).$$

Intuitively, ψ states that no matter which accessible state we choose, we can move to it, eliminate it from the memory, and move to an already remembered state which is connected to some (possibly different) remembered state. Now let

$$\varphi = \textcircled{r}\langle r\rangle(\neg\textcircled{k} \wedge \textcircled{r}\psi).$$

It is clear that $\mathcal{M}_2, 0 \models \varphi$, since one can remember the state 0, then move to state 1 (which is not remembered), and remember it leaving the model in the state $\mathcal{M}_2[0,1]$ and the evaluation state in 1. Then it is easy to see that $\mathcal{M}_2[0,1], 1 \models \psi$. However, one can verify that $\mathcal{M}_1, 0 \not\models \varphi$. Indeed, suppose that, after remembering the state 0, we move to state $n > 0$ and we remember it. By the definition of $\mathcal{M}_1$, the state n will not be reflexive. Now, $\mathcal{M}_1[0,n], n \not\models \psi$ because $\mathcal{M}_1[0,n], 0 \not\models \textcircled{f}(\langle r\rangle(\textcircled{k} \wedge \langle r\rangle\textcircled{k}))$, i.e., $\mathcal{M}_1[n], 0 \not\models \langle r\rangle(\textcircled{k} \wedge \langle r\rangle\textcircled{k})$.

Showing that $\mathcal{ML}_\emptyset(\langle r\rangle) \neq \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e})$ is easier. Let $\varphi = \textcircled{r}\langle r\rangle(\neg\textcircled{k} \wedge \textcircled{e}\textcircled{r}\langle r\rangle\textcircled{k})$. It is not difficult to see that $\mathcal{M}_2, 0 \models \varphi$ but $\mathcal{M}_1, 0 \not\models \varphi$. □

On the other hand, we are still below the expressive power of $\mathcal{HL}(\downarrow)$:

Theorem 18. *$\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e}, \textcircled{f}) \leq \mathcal{HL}(\downarrow)$.*

Proof. In line with the proof of Theorem 10, we define a truth-preserving translation from formulas of $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e}, \textcircled{f})$ into formulas of $\mathcal{HL}(\downarrow)$. To define our translation we use a finite sequence S of nominals in NOM, where each nominal i in the sequence is tagged with a superscript r (representing a remember) or with a superscript f (representing a forget). We use the operation $S \circ i$ to denote the operation of inserting the element i at the end of the sequence S. λ stands for the empty sequence.

$$\begin{array}{rcll}
\mathsf{Tr}_S(p) & = & p \quad p \in \text{PROP} & \\
\mathsf{Tr}_S(\neg\varphi) & = & \neg\mathsf{Tr}_S(\varphi) & \\
\mathsf{Tr}_S(\varphi_1 \wedge \varphi_2) & = & \mathsf{Tr}_S(\varphi_1) \wedge \mathsf{Tr}_S(\varphi_2) & \\
\mathsf{Tr}_S(\langle r\rangle\varphi) & = & \langle r\rangle\mathsf{Tr}_S(\varphi) & \\
\mathsf{Tr}_S(\textcircled{r}\varphi) & = & \downarrow i.\mathsf{Tr}_{S\circ\{i^r\}}(\varphi) & \text{where } i \notin S. \\
\mathsf{Tr}_S(\textcircled{f}\varphi) & = & \downarrow i.\mathsf{Tr}_{S\circ\{i^f\}}(\varphi) & \text{where } i \notin S. \\
\mathsf{Tr}_S(\textcircled{e}\varphi) & = & \mathsf{Tr}_\lambda(\varphi) & \\
\mathsf{Tr}_S(\textcircled{k}) & = & T(S), &
\end{array}$$

where T is a translation from sequences of nominals to $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e}, \textcircled{f})$-formulas defined in the following way:

$$\begin{array}{rcl}
T(\lambda) & = & \bot \\
T(S \circ i^r) & = & i \vee T(S) \\
T(S \circ i^f) & = & \neg i \wedge T(S).
\end{array}$$

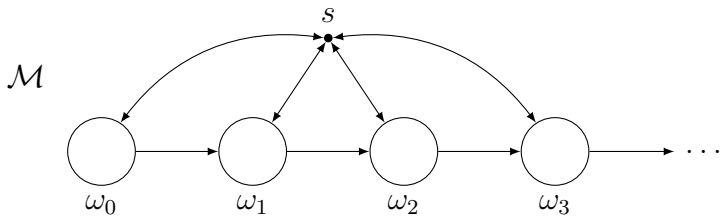
A simple induction shows that $\mathcal{M}, w \models \varphi$ iff $\mathcal{M}, g, w \models \mathsf{Tr}_\lambda(\varphi)$, for any g. □

Corollary 19. *$\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e}) \leq \mathcal{HL}(\downarrow)$ and $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{f}) \leq \mathcal{HL}(\downarrow)$.*

Now we show that $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e})$ is not more expressive than $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{f})$ using a game argument as we did for $\mathcal{ML}_\emptyset(\langle r\rangle)$.

Theorem 20. $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{f}) \not\leq \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e})$.

Proof. Let $\mathcal{M} = \langle \{s\} \cup \omega_0 \cup \omega_1 \cup \ldots, \mathcal{I}, \emptyset\rangle$, where each ω_i is a different copy of ω, and $\mathcal{I}(r) = \{(n, m) \mid n \in \omega_i, m \in \omega_j, i \leq j\} \cup \{(n, s), (s, n) \mid \text{for all } n \neq s\}$, and $\mathcal{I}(p) = \emptyset$ for $p \in$ PROP. Intuitively, the model is as follows



Each ω_i is a total relation on the natural numbers and, in addition, all elements of w_i are related to all elements of w_j if $i < j$.

We prove that $\langle \mathcal{M}, w_0\rangle \equiv^{EF} \langle \mathcal{M}, w_1\rangle$ for $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e})$, where $w_0 \in \omega_0$ and $w_1 \in \omega_1$. Given a state w, we define the *neighborhood* of w as $N(w) = \{v \mid (w, v) \in \mathcal{I}(r)\}$, and we say that the neighborhood of a state w *is remembered* when $N(w) \cap M \neq \emptyset$, where M is the current memory. The strategy we are going to define observes the following invariant:

1. Every time Spoiler has moved to a state w, then Duplicator has answered with an agreeing state v such that $N(w)$ was not remembered if and only if $N(v)$ was not remembered.

2. Every time Spoiler has moved to a state $w \in \omega_i$, Duplicator has answered with a state $v \in \omega_j$. And every time Spoilers has moved to s, Duplicator has moved to s.

It is clear that this invariant holds at the beginning of the game. We will prove that each step of the strategy preserves the invariant. Remember that at any stage of the game, the number of remembered states is always finite. Assume that Spoiler is in a state $w \in \omega_s$ and Duplicator in $v \in \omega_d$. The strategy for Duplicator is the following:

1. If Spoiler decides to remember w, then the game continues with both w and v remembered. So both $N(w)$ and $N(v)$ become remembered, and the invariant is preserved.

2. If Spoiler decides to forget all the states in the model, then both $N(w)$ and $N(v)$ become not remembered, and the game continues with the invariant preserved.

3. If Spoiler moves to s in one model, Duplicator moves to s in the other model. Since every state of every ω_i is connected to s, this is always a possible move for Duplicator. Given the invariant and the fact that s is connected with every other state in the model, it is easy to see that $N(s)$ is not remembered in one model iff $N(s)$ is not remembered in the other model.

4. If Spoiler plays in a state $w' \in \omega_{s'}$ such that $N(w')$ is not remembered, then Duplicator chooses $\omega_{d'}$ and a state $v' \in \omega_{d'}$ such that $N(v')$ is not remembered. Note that by definition of neighborhood and the fact that the accessibility relation is reflexive on ω_i, w' and v' are not in the current memory M. Furthermore, this is always a valid move for Duplicator, given that there are infinitely many ω_i connected with ω_d and the fact that the number of remembered states is finite. So Duplicator can always choose a sufficiently large d' and a state $v' \in \omega_{d'}$ such that $N(v')$ is not remembered.

5. If Spoiler plays in a state $w' \in \omega_{s'}$ such that $N(w')$ is remembered, then Spoiler moves to an agreeing state $v' \in \omega_{d'}$. Let us see that there is always such v' and that the invariant is preserved. If $N(w)$ is not remembered, given the shape of the model, the only possibility is that $w' = s$. Therefore $v' = s$, and we have already seen that the neighborhoods match in this case. The remaining case is when $N(w)$ is remembered. Given the invariant, we know that $N(v)$ is remembered, so if Spoiler chooses $w' \in M$, we know that there is a $v' \in M$ that Spoiler can move to. In this case it is trivial to see that $N(v')$ is not remembered. On the other hand, if Spoiler chooses $w' \notin M$, then a safe choice for Duplicator is a non remembered $v' \in \omega_d$, that is, a state in the same cluster as v. Since each ω_i is infinite, there is always such a v', and also this choice guarantees that $N(v')$ is not remembered.

On the other hand, let $\varphi = \textcircled{r}\langle r\rangle\langle r\rangle(\neg\textcircled{k} \wedge \langle r\rangle\textcircled{k} \wedge \textcircled{r}\langle r\rangle(\textcircled{k} \wedge \textcircled{f}[r]\neg\textcircled{k}))$ be a formula of $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{f})$. It is easy to see that $\mathcal{M}, w_1 \models \varphi$ but $\mathcal{M}, w_0 \not\models \varphi$. □

Corollary 21. *$\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e}) < \mathcal{HL}(\downarrow)$ and $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e}) < \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e}, \textcircled{f})$*

Proof. Trivial given Theorems 20 and 18 □

To end this subsection we want to observe that there are still some interesting questions that remain open. For example, the relation between $\mathcal{HL}(\downarrow)$ and $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e}, \textcircled{f})$:

Question 1. *$\mathcal{HL}(\downarrow) \neq \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e}, \textcircled{f})$?*

We conjecture that the answer is positive, but we have not found yet a pair of models (similarly to the proofs of Theorems 20 and 10) in which the difference can be shown. The other natural question is the relation between $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e})$ and $\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{f})$:

Question 2. *$\mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{e}) \not\leq \mathcal{ML}_\emptyset(\langle r\rangle, \textcircled{f})$?*

Again we conjecture that the answer is positive, and the proof should follow the style of the proof of Theorem 20

4.3 Memory Logics with a Stack

In this subsection we want to analyze other memory containers different than a set. A priori, any kind of data structure could be a suitable alternative, but it should be clear that certain choices immediately gives back the full expressive power of $\mathcal{HL}(\downarrow)$. For example,

suppose we use an unbounded array to store elements of the domain, and that we combine it with suitable operators that can store and retrieve elements to and from a given index. The results is nothing other than a different formulation of assignments and the combination of the $\downarrow$ operator and nominals.

In what follows we will discuss a more subtle case. Suppose that we use a *stack* instead. I.e., our memory structure will still be unbounded, but we are only allowed to store element at the top, and inspect and remove only the top element. We will show that, even though the access to the memory structure is restricted, we still have the full expressive power of $\mathcal{HL}(\downarrow)$. But let us start by formally introducing this language. Its syntax is defined as follows:

$$\text{FORMS} ::= \top \mid p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \langle r\rangle\varphi \mid (\mathtt{push})\varphi \mid (\mathtt{pop})\varphi \mid (\mathtt{top})$$

where $p \in \text{PROP}$, $r \in \text{REL}$ and $\varphi, \varphi_1, \varphi_2 \in \text{FORMS}$. The semantic rules for the new operators are as follows. Assume that S is a stack represented as a list. The symbol λ represent the empty stack, and if S is a stack and w an element $S \cdot w$ represent the stack obtained by adding w as top-most element.

$$\begin{array}{lll}
\langle D, \mathcal{I}, S\rangle, w \models (\mathtt{push})\varphi & \text{iff} & \langle D, \mathcal{I}, S \cdot w\rangle, w \models \varphi \\
\langle D, \mathcal{I}, S \cdot v\rangle, w \models (\mathtt{pop})\varphi & \text{iff} & \langle D, \mathcal{I}, S\rangle, w \models \varphi \\
\langle D, \mathcal{I}, \lambda\rangle, w \models (\mathtt{pop})\varphi & \text{iff} & \text{never} \\
\langle D, \mathcal{I}, S \cdot v\rangle, w \models (\mathtt{top}) & \text{iff} & v = w \\
\langle D, \mathcal{I}, \lambda\rangle, w \models (\mathtt{top}) & \text{iff} & \text{never}
\end{array}$$

Let us call $\mathcal{ML}^{st}_{\emptyset}(\langle r\rangle)$ the logic obtained by adding these operators to the basic modal logic, and when we restrict ourselves to the class where stacks are initially empty. We will show that $\mathcal{ML}^{st}_{\emptyset}(\langle r\rangle)$ and $\mathcal{HL}(\downarrow)$ are equally expressive. Because we are restricting ourselves to the class of models where the stack is initially empty, models in $\mathcal{ML}^{st}_{\emptyset}(\langle r\rangle)$ can be seen as models of $\mathcal{HL}(\downarrow)$ by ignoring the stack.

Theorem 22. $\mathcal{ML}^{st}_{\emptyset}(\langle r\rangle) = \mathcal{HL}(\downarrow)$.

Proof. To prove $\mathcal{ML}^{st}_{\emptyset}(\langle r\rangle) \leq \mathcal{HL}(\downarrow)$, we define a translation mapping a formula in $\mathcal{ML}^{st}_{\emptyset}(\langle r\rangle)$ and a list of nominals N into a formula of $\mathcal{HL}(\downarrow)$.

$$\begin{array}{rcl}
\mathsf{Tr}_N(p) & = & p \quad p \in \text{PROP} \\
\mathsf{Tr}_N(\neg\varphi) & = & \neg\mathsf{Tr}_N(\varphi) \\
\mathsf{Tr}_N(\langle r\rangle\varphi) & = & \langle r\rangle\mathsf{Tr}_N(\varphi) \\
\mathsf{Tr}_N(\varphi_1 \wedge \varphi_2) & = & \mathsf{Tr}_N(\varphi_1) \wedge \mathsf{Tr}_N(\varphi_2) \\
\mathsf{Tr}_N((\mathtt{push})\varphi) & = & \downarrow i.\mathsf{Tr}_{N\cdot i}(\varphi) \quad \text{where } i \notin N \\
\mathsf{Tr}_{N\cdot i}((\mathtt{pop})\varphi) & = & \mathsf{Tr}_N(\varphi) \\
\mathsf{Tr}_{\lambda}((\mathtt{pop})\varphi) & = & \bot \\
\mathsf{Tr}_{N\cdot i}((\mathtt{top})) & = & i \\
\mathsf{Tr}_{\lambda}((\mathtt{top})) & = & \bot
\end{array}$$

We can show by induction in φ that $\mathcal{M}, w \models \varphi$ iff $\mathcal{M}, g, w \models \mathsf{Tr}_{\lambda}(\varphi)$, for any g.

To prove $\mathcal{HL}(\downarrow) \leq \mathcal{ML}^{st}_{\emptyset}(\langle r \rangle)$ we define a translation mapping an $\mathcal{HL}(\downarrow)$-formula and a list of nominals N into an $\mathcal{ML}^{st}_{\emptyset}(\langle r \rangle)$-formula. The translation coincides with the translation above for the propositional, negation, conjunction and modality cases. We translate $\downarrow$ and nominals as follows:

$$\begin{aligned} \mathsf{Tr}_N(\downarrow i.\varphi) &= (\mathtt{push})\mathsf{Tr}_{N \cdot i}(\varphi) \\ \mathsf{Tr}_N(i) &= (\mathtt{pop})^{|N|-n}(\mathtt{top}) \quad i \in \text{NOM}, N[n] = i, \forall m > n : N[m] \neq i, \end{aligned}$$

where $|N|$ represents the length of N and $N[n]$ represents the n-th element of N. It can be shown by induction in φ that if φ is an $\mathcal{HL}(\downarrow)$-sentence, $\mathcal{M}, g, w \models \varphi$ iff $\mathcal{M}, w \models \mathsf{Tr}_\lambda(\varphi)$ for any g. □

4.4 Non empty memory classes

Comparing the expressive power between the logics that start evaluating formulas in $\mathcal{C}_\emptyset$ and the ones that use an arbitrary memory poses a complication because, strictly speaking, each of them uses a different class of models. In this case it is not as obvious how to define the mapping between each type of models. The most natural option seems to involve a shift in the signature of the language, in order to preserve the information stored in the models.

Consider, for example, an arbitrary model $\mathcal{M} = \langle D, \mathcal{I}, M \rangle$ for $\mathcal{ML}(\langle r \rangle)$. If we want to consider it as a model of $\mathcal{ML}_\emptyset(\langle r \rangle)$ we need to 'make room' for the non empty memory M somehow. We will do that by considering $\mathcal{M} = \langle D, \mathcal{I}, M \rangle$ as a model over a signature with one additional propositional symbol which we will call *known* and that will be interpreted as M.

Theorem 23.

1. $\mathcal{ML}_\emptyset(\langle r \rangle)$ *over the signature* $\langle \text{PROP} \cup \{known\}, \text{REL} \rangle$ *is equivalent to* $\mathcal{ML}(\langle r \rangle)$ *over the signature* $\langle \text{PROP}, \text{REL} \rangle$.

2. $\mathcal{ML}_\emptyset(\langle\langle r \rangle\rangle)$ *over the signature* $\langle \text{PROP} \cup \{known\}, \text{REL} \rangle$ *is equivalent to* $\mathcal{ML}(\langle\langle r \rangle\rangle)$ *over the signature* $\langle \text{PROP}, \text{REL} \rangle$.

Proof. The argument for 2 is exactly the same as the one for 1. Hence, let us prove $\mathcal{ML}_\emptyset(\langle r \rangle) = \mathcal{ML}(\langle r \rangle)$ (over the appropriate signatures).

We start by associating every model $\mathcal{M} = \langle D, \mathcal{I}, M \rangle$ of $\mathcal{ML}(\langle r \rangle)$ over the signature $\langle \text{PROP}, \text{REL} \rangle$ with the model $\mathcal{M}' = \langle D, \mathcal{I}', \emptyset \rangle$ of $\mathcal{ML}_\emptyset(\langle r \rangle)$ over the signature $\langle \text{PROP} \cup \{known\}, \text{REL} \rangle$ where $\mathcal{I}'$ is identical to $\mathcal{I}$ over PROP and REL and $\mathcal{I}'(known) = M$.

$[\mathcal{ML}_\emptyset(\langle r \rangle) \leq \mathcal{ML}(\langle r \rangle)]$: use the translation Tr that replaces occurrences of the propositional symbol *known* by Ⓚ in any formula of $\mathcal{ML}_\emptyset(\langle r \rangle)$. Clearly for any formula $\varphi \in \mathcal{ML}_\emptyset(\langle r \rangle)$ we have that $\mathcal{M}', w \models \varphi$ iff $\mathcal{M}, w \models \mathsf{Tr}(\varphi)$.

$[\mathcal{ML}(\langle r \rangle) \leq \mathcal{ML}_\emptyset(\langle r \rangle)]$: use the translation Tr that replaces occurrences of Ⓚ by $(Ⓚ \vee known)$ in any formula of $\mathcal{ML}(\langle r \rangle)$. Clearly for any formula $\varphi \in \mathcal{ML}(\langle r \rangle)$ we have that $\mathcal{M}, w \models \varphi$ iff $\mathcal{M}', w \models \mathsf{Tr}(\varphi)$. □

Intuitively, the only thing we need to do is to store $\mathcal{I}(known)$ in the starting memory and vice versa. In the presence of Ⓕ and Ⓔ the memory does not grow monotonically and hence we cannot simulate it using $known \vee$ Ⓚ.

Now, the same expressivity hierarchy we proved for logics with empty memory can be established for logics with arbitrary memory, that is $\mathcal{K} < \mathcal{ML}(\langle\!\langle r \rangle\!\rangle) < \mathcal{ML}(\langle r \rangle) < \mathcal{HL}(\downarrow)$:

Theorem 24.

1. *$\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ over the signature $\langle \text{PROP}, \text{REL} \rangle$ is strictly more expressive than $\mathcal{K}$ over the signature $\langle \text{PROP} \cup \{known\}, \text{REL} \rangle$.*
2. *$\mathcal{ML}(\langle\!\langle r \rangle\!\rangle) < \mathcal{ML}(\langle r \rangle)$.*
3. *$\mathcal{HL}(\downarrow)$ over the signature $\langle \text{PROP} \cup \{known\}, \text{REL}, \text{NOM} \rangle$ is strictly more expressive than $\mathcal{ML}(\langle r \rangle)$ over the signature $\langle \text{PROP}, \text{REL} \rangle$.*

Proof. The proof for 1 is the same as the proof for Theorem 9 The proof for 2 is the same as the proof for Theorem 8 To prove 3 we adapt the translation Tr defined in the proof of Theorem 10 with the following clause for Ⓚ:

$$\mathsf{Tr}_N(\text{Ⓚ}) \;=\; \big(\bigvee_{i \in N} i\big) \vee known.$$

$\mathcal{HL}(\downarrow) \not\leq \mathcal{ML}(\langle r \rangle)$ can then be shown using the following models. Let $\mathcal{M}_1 = \langle \{w\}, \mathcal{I}_1, \{w\} \rangle$ with $\mathcal{I}_1(r) = \{(w, w)\}$ and $\mathcal{I}_1(p) = \emptyset$ for $p \in \text{PROP}$; and $\mathcal{M}_2 = \langle \{u, v\}, \mathcal{I}_2, \{u, v\} \rangle$ with $\mathcal{I}_2(r) = \{(u, v), (v, u)\}$ $\mathcal{I}_2(p) = \emptyset$ for $p \in \text{PROP}$.

Duplicator always wins on $EF(\mathcal{M}_1, \mathcal{M}_2, w, u)$ and thus $\mathcal{M}_1, w \equiv^{EF}_{\mathcal{ML}(\langle r \rangle)} \mathcal{M}_2, u$. On the other hand, $\mathcal{M}_1, w \models \downarrow i.\langle r \rangle i$ but $\mathcal{M}_2, u \not\models \downarrow i.\langle r \rangle i$. □

5 (Un)Decidability and the Finite Model Property

In the previous section we showed memory logics more expressive than $\mathcal{K}$ but less expressive than $\mathcal{HL}(\downarrow)$ (the only exception is $\mathcal{ML}^{st}_{\emptyset}(\langle r \rangle)$, which has the same expressive power than $\mathcal{HL}(\downarrow)$). Given that $\mathcal{K}$ is decidable and $\mathcal{HL}(\downarrow)$ undecidable (Areces & ten Cate, 2006), exploring where the decidability line lies is an intriguing question. The main goal of this section is to investigate this issue, together with the related question of whether the logic is sufficiently expressive to force infinite models.

We start by investigating $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ and $\mathcal{ML}_{\emptyset}(\langle\!\langle r \rangle\!\rangle)$. We will show that even though they are equivalent in terms of expressive power when we allow a shift in the signature, the satisfiability problem for $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ is decidable (actually PSPACE-complete) while $\mathcal{ML}_{\emptyset}(\langle\!\langle r \rangle\!\rangle)$ is already undecidable. As we will show in the proof of Theorem 27 the trick is to use a 'dirty' memory. In $\mathcal{ML}_{\emptyset}(\langle\!\langle r \rangle\!\rangle)$, we are restricted to the class of models where the memory is always initialized to $\emptyset$ and we can't play this trick anymore. Actually $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ is really standing on the decidability line: adding a single nominal to $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ pushes the satisfiability problem over to undecidability.

5.1 The Decidability of $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$

We will first prove that $\mathcal{K}$ and $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ are expressively equivalent over the class of tree models. We will then prove that $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ has a tree model property. With those results at hand, decidability and PSPACE-completeness of $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ easily follows.

Theorem 25. *Restricted to the class of tree models, the logic $\mathcal{K}$ over the signature $\langle \text{PROP} \cup \{known\}, \text{REL} \rangle$ is equivalent to $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ over the signature $\langle \text{PROP}, \text{REL} \rangle$.*

Proof. [$\mathcal{K} \leq \mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$]: This is a direct corollary of Theorem 9
[$\mathcal{ML}(\langle\!\langle r \rangle\!\rangle) \leq \mathcal{K}$]: We start by noticing that in $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ we can eliminate Ⓚ at modal depth 0 from a formula like Ⓡφ.

Claim: Let $\varphi^\sharp$ be the result of replacing all the occurrences of Ⓚ that are in $\varphi \in \mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ at modal depth zero by $\top$. Then $\mathcal{M}, w \models$ Ⓡφ if and only if $\mathcal{M}, w \models \varphi^\sharp$.

Proof of Claim. We proceed by induction on φ. The case for Ⓚ, the propositional symbols and booleans are straightforward. We analyze the other cases:

- $\varphi =$ Ⓡψ. $\mathcal{M}, w \models$ ⓇⓇψ iff $\mathcal{M}, w \models$ Ⓡψ iff (by inductive hypothesis) $\mathcal{M}, w \models \psi^\sharp$ iff $\mathcal{M}, w \models (\psi^\sharp)^\sharp$ iff (by inductive hypothesis) $\mathcal{M}, w \models$ Ⓡ$(\psi^\sharp)$ iff $\mathcal{M}, w \models ($Ⓡ$\psi)^\sharp$.
- $\varphi = \langle\!\langle r \rangle\!\rangle\psi$. $\mathcal{M}, w \models$ Ⓡ$\langle\!\langle r \rangle\!\rangle\psi$ iff (by definition) $\mathcal{M}[w], w \models \langle\!\langle r \rangle\!\rangle\psi$ iff (by definition of $\sharp$) $\mathcal{M}[w], w \models (\langle\!\langle r \rangle\!\rangle\psi)^\sharp$ iff (by definition) $\mathcal{M}, w \models (\langle\!\langle r \rangle\!\rangle\psi)^\sharp$. ⊣

Define now the following translation taking $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$-formulas over the signature $\langle \text{PROP}, \text{REL} \rangle$ to $\mathcal{K}$-formulas over the signature $\langle \text{PROP} \cup \{known\}, \text{REL} \rangle$:

$$\begin{array}{rcl} \mathsf{Tr}(p) & = & p \quad p \in \text{PROP} \\ \mathsf{Tr}(\text{Ⓚ}) & = & known \\ \mathsf{Tr}(\neg\varphi) & = & \neg\mathsf{Tr}(\varphi) \\ \mathsf{Tr}(\varphi_1 \wedge \varphi_2) & = & \mathsf{Tr}(\varphi_1) \wedge \mathsf{Tr}(\varphi_2) \\ \mathsf{Tr}(\langle\!\langle r \rangle\!\rangle\varphi) & = & \langle r \rangle \mathsf{Tr}(\varphi) \\ \mathsf{Tr}(\text{Ⓡ}\varphi) & = & \mathsf{Tr}(\varphi^\sharp). \end{array}$$

Let $\varphi \in \mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$, and let $\mathcal{M} = \langle D, \mathcal{I}, M \rangle$ be an arbitrary tree model. Let $\mathcal{M}' = \langle D, \mathcal{I}' \rangle$ where $\mathcal{I}'$ is identical to $\mathcal{I}$ except that $\mathcal{I}'(known) = M$. We can prove that $\mathcal{M}, w \models \varphi$ if and only if $\mathcal{M}', w \models \mathsf{Tr}(\varphi)$.

We proceed by induction on φ. The propositional and boolean cases are trivial. The Ⓚ case is also easy given the definitions. Let us consider $\varphi = \langle\!\langle r \rangle\!\rangle\psi$. Because $\mathcal{M}$ is a tree, the remember operator has no effect beyond modal operators, so $\mathcal{M}, w \models \langle\!\langle r \rangle\!\rangle\psi$ if and only if there exists v such that $(w, v) \in \mathcal{I}(r)$ and $\mathcal{M}, v \models \psi$. By inductive hypothesis, $\mathcal{M}', v \models \psi$ iff $\mathcal{M}', v \models \mathsf{Tr}(\psi)$, and by definition $\mathcal{M}', w \models \langle r \rangle \mathsf{Tr}(\psi)$. Finally, let us see the case for remember. By the previous Claim, $\mathcal{M}, w \models$ Ⓡψ iff $\mathcal{M}, w \models \psi^\sharp$. By inductive hypothesis, $\mathcal{M}, w \models \mathsf{Tr}(\psi^\sharp)$. □

We now prove that $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ has the *tree model property* (Blackburn et al., 2001), that is, every satisfiable formula in $\mathcal{ML}(\langle\!\langle r \rangle\!\rangle)$ is satisfied in a tree model.

Theorem 26 (Tree model property)**.** *Let $\langle \mathcal{M}, w\rangle$ be a $\mathcal{ML}(\langle\!\langle r\rangle\!\rangle)$-model. Then there is a tree $\mathcal{M}'$ such that $\langle \mathcal{M}, w\rangle \equiv^{EF} \langle \mathcal{M}', w\rangle$.*

Proof. We prove the result for the unimodal case, the generalization to the multimodal case is straightforward. Let $\mathcal{M} = \langle D, \mathcal{I}, M\rangle$, define $\mathcal{M}' = \langle D', \mathcal{I}', M'\rangle$ as follows. Its domain D' consists of all finite sequences $\bar{u} = (u_0, \ldots, u_n)$ such that $u_0 = w$, $n \geq 0$ and $(u_i, u_{i+1}) \in \mathcal{I}(r)$ for $0 \leq i < n$. Let $\bar{u} = (u_0, \ldots, u_n)$ and $\bar{v} = (v_0, \ldots, v_m)$, then define $\mathcal{I}'(r)$ as follows, $(\bar{u}, \bar{v}) \in \mathcal{I}'(r)$ if and only if $m = n + 1$, $u_i = v_1$ for $i = 0, \ldots, n$ and $(u_n, v_m) \in \mathcal{I}(r)$. $\mathcal{I}'(p)$ is defined by setting $(u_0, \ldots, u_n) \in \mathcal{I}'(p)$ iff $u_n \in \mathcal{I}(p)$. Finally, $(u_0, \ldots, u_{n-1}, u_n) \in M'$ iff $u_n \in \{u_0, \ldots, u_{n-1}\}$ or $u_n \in M$.

Let s_i be the sequence $(v_0, \ldots, v_i)$. We show that Duplicator has a winning strategy in the game $EF(\mathcal{M}, \mathcal{M}', w, w)$. It is sufficient to see that in the game

$$EF(\mathcal{M}[v_0, \ldots, v_n], \mathcal{M}'[s_0, \ldots, s_n], v_{n+1}, s_{n+1}),$$

Duplicator can always answer successfully to Spoiler's moves.

- If Spoiler chooses $\mathcal{M}[v_0, \ldots, v_n]$ and some v_{n+1}, a successor of v_n, Duplicator chooses the sequence $s_{n+1} = s_n v_{n+1}$.
- If Spoiler chooses $\mathcal{M}'[s_0, \ldots, s_n]$ and $s_{n+1} = s_n v_{n+1}$ (for some v_{n+1}), a successor of s_n, Duplicator chooses the state v_{n+1}.

By definition s_{n+1} and v_{n+1} agree. Observe that the memory of $\mathcal{M}[v_0, \ldots, v_n]$ is $M \cup \{v_0, \ldots, v_n\}$ and the memory of $\mathcal{M}'[s_0, \ldots, s_n]$ is $M' \cup \{s_0, \ldots, s_n\}$. It is also clear that $v_{n+1} \in M$ if and only if $s_{n+1} \in M'$. Formally, $v_{n+1} \in M \cup \{v_0, \ldots, v_n\}$ implies $s_{n+1} \in M'$ by definition. And $s_{n+1} \in S' \cup \{s_0, \ldots, s_n\}$, then $s_{n+1} \in S'$ (since there are no cycles in $\mathcal{M}'$) and by definition $v_{n+1} \in M \cup \{v_0, \ldots, v_n\}$. □

Theorem 27. *The satisfiability problem of $\mathcal{ML}(\langle\!\langle r\rangle\!\rangle)$ is* PSPACE*-complete.*

Proof. We first show decidability of the satisfiability problem of $\mathcal{ML}(\langle\!\langle r\rangle\!\rangle)$ proving that any satisfiable formula of $\mathcal{ML}(\langle\!\langle r\rangle\!\rangle)$ is satisfiable in a recursively bounded model. Let φ be a $\mathcal{ML}(\langle\!\langle r\rangle\!\rangle)$-formula of modal depth k, and suppose $\mathcal{M}_1, w \models \varphi$. By Theorem 26, there is a tree $\mathcal{M}_2$ such that $\mathcal{M}_2, w \models \varphi$. Using Theorem 25, we know that $\mathcal{M}_2, w \models \mathsf{Tr}(\varphi)$ (here, $\mathcal{M}_2$ is taken as a $\mathcal{K}$ model over the appropriate signature). Now we can use the *bounded* tree model property for basic modal logic (Blackburn et al., 2001), so there must be a recursively bounded tree $\mathcal{M}_3 = \langle D_3, \mathcal{I}_3\rangle$ and $v \in D_3$ such that $\mathcal{M}_3, v \models \mathsf{Tr}(\varphi)$. Finally, we can use Theorem 25 again, and conclude $\langle D_3, \mathcal{I}_3', \mathcal{I}_3(\mathit{known})\rangle, v \models \varphi$, where $\mathcal{I}_3'$ is the restriction of $\mathcal{I}_3$ to the signature that does not contain *known*.

The PSPACE-completeness follows from the fact that the translation Tr is linear, and that the satisfiability problem for the basic modal logic is PSPACE (Blackburn et al., 2001). □

Now we show that adding Ⓕ to $\mathcal{ML}(\langle\!\langle r\rangle\!\rangle)$ keeps the logic decidable. In fact, $\mathcal{ML}(\langle\!\langle r\rangle\!\rangle, \text{Ⓕ})$ can be encoded into $\mathcal{ML}(\langle\!\langle r\rangle\!\rangle)$ using a linear translation. In other words, $\mathcal{ML}(\langle\!\langle r\rangle\!\rangle, \text{Ⓕ}) \leq \mathcal{ML}(\langle\!\langle r\rangle\!\rangle)$ and hence Ⓕ does not add expressivity when added to $\mathcal{ML}(\langle\!\langle r\rangle\!\rangle)$. At the end of this section we are going to see that this is not the case for Ⓔ.

Theorem 28. *The satisfiability problem for* $\mathcal{ML}(\langle\langle r\rangle\rangle, \textcircled{\texttt{f}})$ *is* PSPACE*-complete.*

Proof. We show that there is a linear translation from $\mathcal{ML}(\langle\langle r\rangle\rangle, \textcircled{\texttt{f}})$ to $\mathcal{ML}(\langle\langle r\rangle\rangle)$. Let Tr_a be the following translation from $\mathcal{ML}(\langle\langle r\rangle\rangle, \textcircled{\texttt{f}})$ formulas to $\mathcal{ML}(\langle\langle r\rangle\rangle)$ formulas, where a ranges over $\{r, f\}$:

$$\begin{array}{rcl}
\mathsf{Tr}_a(p) & = & p \quad p \in \text{PROP} \\
\mathsf{Tr}_r(\textcircled{\texttt{k}}) & = & \textcircled{\texttt{k}} \\
\mathsf{Tr}_f(\textcircled{\texttt{k}}) & = & \neg\top \\
\mathsf{Tr}_a(\neg\varphi) & = & \neg\mathsf{Tr}_a(\varphi) \\
\mathsf{Tr}_a(\varphi_1 \wedge \varphi_2) & = & \mathsf{Tr}_a(\varphi_1) \wedge \mathsf{Tr}_a(\varphi_2) \\
\mathsf{Tr}_a(\langle\langle r\rangle\rangle\varphi) & = & \langle\langle r\rangle\rangle\mathsf{Tr}_r(\varphi) \\
\mathsf{Tr}_a(\textcircled{\texttt{r}}\varphi) & = & \textcircled{\texttt{r}}\mathsf{Tr}_r(\varphi) \\
\mathsf{Tr}_a(\textcircled{\texttt{f}}\varphi) & = & \mathsf{Tr}_f(\varphi).
\end{array}$$

Given a model $\mathcal{M} = \langle D, \mathcal{I}, M\rangle$ and a state $w \in D$, we define $\mathcal{M}[\text{-}w] = \langle D, \mathcal{I}, M \setminus \{w\}\rangle$. We prove by mutual induction on φ these two properties:

(1) $\mathcal{M}, w \models \varphi$ iff $\mathcal{M}, w \models \mathsf{Tr}_r(\varphi)$.

(2) $\mathcal{M}[\text{-}w], w \models \varphi$ iff $\mathcal{M}, w \models \mathsf{Tr}_f(\varphi)$.

Notice that (1) in fact shows that $\mathcal{ML}(\langle\langle r\rangle\rangle, \textcircled{\texttt{f}}) \leq \mathcal{ML}(\langle\langle r\rangle\rangle)$. The only interesting cases for both properties are $\langle\langle r\rangle\rangle$, $\textcircled{\texttt{r}}$ and $\textcircled{\texttt{f}}$. For the property (1), let $\varphi = \langle\langle r\rangle\rangle\psi$. $\mathcal{M}, w \models \langle\langle r\rangle\rangle\psi$ iff (by definition) there is a $w' \in D$, such that $(w, w') \in \mathcal{I}(r)$ and $\mathcal{M}[w], w' \models \psi$ iff (by inductive hypothesis on (i)) $\mathcal{M}[w], w' \models \mathsf{Tr}_r(\psi)$ iff (by definition) $\mathcal{M}, w \models \langle\langle r\rangle\rangle\mathsf{Tr}_r(\psi)$ iff (by definition) $\mathcal{M}, w \models \mathsf{Tr}_r(\langle\langle r\rangle\rangle\psi)$. The next case is $\varphi = \textcircled{\texttt{r}}\psi$. $\mathcal{M}, w \models \textcircled{\texttt{r}}\psi$ iff (by definition) $\mathcal{M}[w], w \models \psi$ iff (by inductive hypothesis on (1)) $\mathcal{M}[w], w \models \mathsf{Tr}_r(\psi)$ iff (by definition) $\mathcal{M}, w \models \textcircled{\texttt{r}}\mathsf{Tr}_r(\psi)$ iff (by definition) $\mathcal{M}, w \models \mathsf{Tr}_r(\textcircled{\texttt{r}}\psi)$. Finally, let $\varphi = \textcircled{\texttt{f}}\psi$. $\mathcal{M}, w \models \textcircled{\texttt{f}}\psi$ iff (by definition) $\mathcal{M}[\text{-}w], w \models \psi$ iff (by inductive hypothesis on (2)) $\mathcal{M}, w \models \mathsf{Tr}_f(\psi)$ iff (by definition) $\mathcal{M}, w \models \mathsf{Tr}_r(\textcircled{\texttt{f}}\psi)$.

For the property (2), let $\varphi = \langle\langle r\rangle\rangle\psi$. $\mathcal{M}[\text{-}w], w \models \langle\langle r\rangle\rangle\psi$ iff (by definition) there is a $w' \in D$, such that $(w, w') \in \mathcal{I}(r)$ and $\mathcal{M}[w], w' \models \psi$ iff (by inductive hypothesis on (1)) $\mathcal{M}[w], w' \models \mathsf{Tr}_r(\psi)$ iff (by definition) $\mathcal{M}, w \models \langle\langle r\rangle\rangle\mathsf{Tr}_r(\psi)$ iff (by definition) $\mathcal{M}, w \models \mathsf{Tr}_f(\langle\langle r\rangle\rangle\psi)$. The next case is $\varphi = \textcircled{\texttt{r}}\psi$. $\mathcal{M}[\text{-}w], w \models \textcircled{\texttt{r}}\psi$ iff (by definition) $\mathcal{M}[w], w \models \psi$ (by inductive hypothesis on (1)) $\mathcal{M}[w], w \models \mathsf{Tr}_r(\psi)$ iff (by definition) $\mathcal{M}, w \models \textcircled{\texttt{r}}\mathsf{Tr}_r(\psi)$ iff (by definition) $\mathcal{M}, w \models \mathsf{Tr}_f(\textcircled{\texttt{r}}\psi)$. The last case is $\varphi = \textcircled{\texttt{f}}\psi$. $\mathcal{M}[\text{-}w], w \models \textcircled{\texttt{f}}\psi$ iff (by definition) $\mathcal{M}[\text{-}w], w \models \psi$ (by inductive hypothesis on (2)) $\mathcal{M}, w \models \mathsf{Tr}_f(\psi)$ iff (by definition) $\mathcal{M}, w \models \mathsf{Tr}_f(\textcircled{\texttt{f}}\psi)$.

Property (1) shows that we have defined a satisfiability preserving translation. Observe that the linearity of the translation is trivial, and given Theorem 27 we conclude the desired result. □

5.2 Undecidable Memory Logics

While $\mathcal{ML}(\langle\langle r \rangle\rangle)$ is decidable, it seems to be standing at the border of undecidability. The logic $\mathcal{ML}_{\emptyset}(\langle\langle r \rangle\rangle)$, obtained from $\mathcal{ML}(\langle\langle r \rangle\rangle)$ by restricting the class of models to those where $S = \emptyset$ is undecidable. Actually, the logic $\mathcal{ML}(\langle\langle r \rangle\rangle) + i$, obtained by adding a single nominal to $\mathcal{ML}(\langle\langle r \rangle\rangle)$, is already undecidable. We first prove failure of the finite model property for $\mathcal{ML}(\langle\langle r \rangle\rangle) + i$.

Theorem 29. *$\mathcal{ML}(\langle\langle r \rangle\rangle) + i$ lacks the finite model property.*

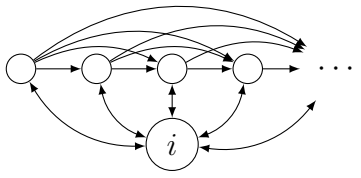
Proof. Consider the following formulas:

$$
\begin{array}{rl}
(\mathit{Back}) & i \wedge [\![r]\!]\neg i \wedge \langle\langle r \rangle\rangle \top \wedge [\![r]\!]\langle\langle r \rangle\rangle i \\
(\mathit{Empty}) & [\![r]\!]\neg \textcircled{\mathtt{k}} \wedge [\![r]\!][\![r]\!](\neg i \rightarrow \neg \textcircled{\mathtt{k}}) \\
(\mathit{Spy}) & [\![r]\!][\![r]\!](\neg i \rightarrow \langle\langle r \rangle\rangle(i \wedge \langle\langle r \rangle\rangle(\textcircled{\mathtt{k}} \wedge \neg\langle\langle r \rangle\rangle(\textcircled{\mathtt{k}} \wedge \neg i)))) \\
(\mathit{Succ}) & [\![r]\!]\langle\langle r \rangle\rangle\neg i \\
(\mathit{No\text{-}3cyc}) & \neg\langle\langle r \rangle\rangle\langle\langle r \rangle\rangle(\neg \textcircled{\mathtt{k}} \wedge \langle\langle r \rangle\rangle(\neg \textcircled{\mathtt{k}} \wedge \langle\langle r \rangle\rangle(\textcircled{\mathtt{k}} \wedge \neg i))) \\
(\mathit{Tran}) & [\![r]\!]\langle\langle r \rangle\rangle(i \wedge [\![r]\!](\neg \textcircled{\mathtt{k}} \rightarrow \langle\langle r \rangle\rangle(i \wedge \langle\langle r \rangle\rangle(\textcircled{\mathtt{k}} \wedge \langle\langle r \rangle\rangle(\textcircled{\mathtt{k}} \wedge \neg i))))).
\end{array}
$$

Let *Inf* be $\mathit{Back} \wedge \mathit{Empty} \wedge \mathit{Spy} \wedge \mathit{Succ} \wedge \mathit{No\text{-}3cyc} \wedge \mathit{Tran}$. Let $\mathcal{M} = \langle D, \mathcal{I}, M \rangle$. We claim that if $\mathcal{M}, w \models \mathit{Inf}$, then D is infinite.

Suppose $\mathcal{M}, w \models \mathit{Inf}$. Let $B = \{b \in D \mid (w, b) \in \mathcal{I}(r)\}$. Because *Back* is satisfied, $w \notin B$, $B \neq \emptyset$ and for all $b \in B$, $(b, w) \in \mathcal{I}(r)$. Note that *Empty* says that the one and two-step neighbors of w are not in M, and this also implies that every state in B is irreflexive. Because *Spy* is satisfied, if $a \neq w$ and a is a successor of an element of B then a is also an element of B. As *Succ* is satisfied at w, every point in B has a successor distinct from w. *No-3cyc* disallow cycles of size 2 or 3 in B; and together with *Tran* they force $\mathcal{I}(r)$ to transitively order B.

It follows that B is an unbounded strict partial order as showed in the picture below, hence infinite, and so is D.



□

We now show that $\mathcal{ML}(\langle\langle r \rangle\rangle) + i$ is undecidable by encoding the $\omega \times \omega$ tiling problem (see Börger et al. (1997)). Following ideas in (Blackburn & Seligman, 1995), we will use three modalities $\langle s \rangle$, $\langle u \rangle$ and $\langle r \rangle$. We construct a *spy point* over the relation $\mathcal{I}(s)$ (i.e., the point of evaluation will have access in one $\mathcal{I}(s)$-step to any reachable state in the model). The relations $\mathcal{I}(u)$ and $\mathcal{I}(r)$ represent moving up and to the right, respectively, from one tile to the other. We code each type of tile with a fixed propositional symbol t_i. With this encoding we define for each tiling problem T, a formula φ^T such that the set of tiles types T tiles $\omega \times \omega$ iff φ^T has a model.

Theorem 30. *The satisfiability problem for $\mathcal{ML}(\langle\langle r \rangle\rangle) + i$ is undecidable.*

Proof. Let $T = \{T_1, \ldots, T_n\}$ be a set of tile types. Given a tile type T_i, $u(T_i)$, $r(T_i)$, $d(T_i)$, $l(T_i)$ will represent the colors of the up, right, down and left edges of T_i respectively. Define:

$$
\begin{array}{rll}
(\mathit{Back}) & i \wedge [\![s]\!]\neg i \wedge \langle\langle s \rangle\rangle \top \wedge [\![s]\!]\langle\langle s \rangle\rangle i \wedge [\![s]\!][\![s]\!] i & \\
(\mathit{Empty}) & [\![s]\!]\neg \textcircled{k} \wedge [\![s]\!][\![\dagger]\!]\neg \textcircled{k} & \dagger \in \{r, u\} \\
(\mathit{Spy}) & [\![s]\!][\![\dagger]\!]\langle\langle s \rangle\rangle (i \wedge \langle\langle s \rangle\rangle (\textcircled{k} \wedge \neg\langle\langle \dagger \rangle\rangle \textcircled{k})) & \dagger \in \{r, u\} \\
(\mathit{Grid}) & [\![s]\!]\langle\langle \dagger \rangle\rangle \top & \dagger \in \{r, u\} \\
(\mathit{Func}) & [\![s]\!][\![\dagger]\!]\langle\langle s \rangle\rangle\langle\langle s \rangle\rangle (\textcircled{k} \wedge \langle\langle \dagger \rangle\rangle \textcircled{k} \wedge [\![\dagger]\!]\textcircled{k}) & \dagger \in \{r, u\} \\
(\mathit{Conf}) & [\![s]\!]\langle\langle u \rangle\rangle\langle\langle r \rangle\rangle\langle\langle s \rangle\rangle\langle\langle s \rangle\rangle (\textcircled{k} \wedge \neg\langle\langle r \rangle\rangle \textcircled{k} \wedge \langle\langle u \rangle\rangle \textcircled{k} \wedge & \\
 & \quad \langle\langle r \rangle\rangle (\neg \textcircled{k} \wedge (\langle\langle u \rangle\rangle (\textcircled{k} \wedge \neg\langle\langle r \rangle\rangle \textcircled{k}))) & \\
(\mathit{UR\text{-}no\text{-}Cycle}) & [\![s]\!][\![u]\!][\![r]\!]\neg \textcircled{k} \wedge [\![s]\!][\![r]\!][\![u]\!]\neg \textcircled{k} & \\
(\mathit{URU\text{-}no\text{-}Cycle}) & [\![s]\!][\![u]\!][\![r]\!][\![u]\!]\neg \textcircled{k} & \\
(\mathit{Unique}) & [\![s]\!] \left(\bigvee_{1 \leq i \leq n} t_i \wedge \bigwedge_{1 \leq i < j \leq n} (t_i \rightarrow \neg t_j) \right) & \\
(\mathit{Vert}) & [\![s]\!] \bigwedge_{1 \leq i \leq n} \left(t_i \rightarrow \langle\langle u \rangle\rangle \bigvee_{1 \leq j \leq n, u(T_i) = d(T_j)} t_j \right) & \\
(\mathit{Horiz}) & [\![s]\!] \bigwedge_{1 \leq i \leq n} \left(t_i \rightarrow \langle\langle r \rangle\rangle \bigvee_{1 \leq j \leq n, r(T_i) = l(T_j)} t_j \right). &
\end{array}
$$

Let the formula φ^T be the conjunction of all the above formulas. We show that T tiles $\omega \times \omega$ if and only if φ^T is satisfiable.

Suppose $\mathcal{M}, w \models \varphi^T$. Observe that *Back* and *Spy*, together with *Empty* make w a spy via $\mathcal{I}(s)$ (and also force $\mathcal{I}(u)$ and $\mathcal{I}(r)$ to be irreflexive and asymmetric). These $\mathcal{I}(s)$-accessible states will represent the tiles. We will have that $[\![s]\!]\psi$ holds at w iff ψ is true at every tile, and $\langle\langle s \rangle\rangle\langle\langle s \rangle\rangle\psi$ holds at tile v iff ψ is true at some (perhaps the same) tile. Now, *Grid* states that from every tile there is another tile moving up (that is, following the $\mathcal{I}(u)$-relation). The same holds for the right direction (following the $\mathcal{I}(r)$-relation). *Func* (together with *Back* and *Spy*) forces $\mathcal{I}(u)$ and $\mathcal{I}(r)$ to be functional. *Conf* ensures that the tiles are arranged as a grid, once we force $\mathcal{I}(u) \circ \mathcal{I}(r)$, the composition of $\mathcal{I}(u)$ and $\mathcal{I}(r)$, to be irreflexive (*UR-No-Cycle*), and we forbid the existence of cycles following successive steps in the $\mathcal{I}(u)$, $\mathcal{I}(r)$ and $\mathcal{I}(u)$ relations, in that order (*URU-no-Cycle*).

That completes the description of the grid. The last three formulas ensure that every tile has a unique type t_i, and that the colors of the tiles match properly. From this, it easily follows that $\mathcal{M}$ is a tiling of $\omega \times \omega$.

For the converse, suppose $f : \omega \times \omega \rightarrow T$ is a tiling of $\omega \times \omega$. We define the model $\mathcal{M} = \langle \omega \times \omega \cup \{w\}, \mathcal{I}, \emptyset \rangle$ where $\mathcal{I}$ is

$$
\begin{array}{l}
\mathcal{I}(s) = \{(w, v), (v, w) \mid v \in \omega \times \omega\} \text{ (hence } w \text{ will act as the spy point)} \\
\mathcal{I}(u) = \{((x, y), (x, y + 1)) \mid x, y \in \omega\} \\
\mathcal{I}(r) = \{((x, y), (x + 1, y)) \mid x, y \in \omega\} \\
\mathcal{I}(p) = \{w\} \\
\mathcal{I}(t_i) = \{x \mid x \in \omega \times \omega, f(x) = T_i\}.
\end{array}
$$

The reader may verify that, by construction, $\mathcal{M}, w \models \varphi^T$. □

We now turn to $\mathcal{ML}_\emptyset(\langle\langle r\rangle\rangle)$. The ideas are similar to the case of $\mathcal{ML}(\langle\langle r\rangle\rangle)+i$ but this time we cannot use the nominal i to make the spy point. On the other hand, we know that the memory is empty when we start evaluating a formula.

Theorem 31. *$\mathcal{ML}_\emptyset(\langle\langle r\rangle\rangle)$ lacks the finite model property.*

Proof. Consider the following formulas:

$$\begin{array}{rl}
(\mathit{Back}) & p \wedge [\![r]\!]\neg p \wedge \langle\langle r\rangle\rangle\top \wedge [\![r]\!]\langle\langle r\rangle\rangle(\textcircled{k} \wedge p)\\
(\mathit{Spy}) & [\![r]\!][\![r]\!](\neg p \rightarrow \langle\langle r\rangle\rangle(\textcircled{k} \wedge p \wedge \langle\langle r\rangle\rangle(\textcircled{k} \wedge \neg\langle\langle r\rangle\rangle(\textcircled{k} \wedge \neg p))))\\
(\mathit{Irr}) & \neg\langle\langle r\rangle\rangle(\neg p \wedge \langle\langle r\rangle\rangle(\neg p \wedge \textcircled{k})))\\
(\mathit{Succ}) & [\![r]\!]\langle\langle r\rangle\rangle\neg p\\
(\mathit{No\text{-}3cyc}) & \neg\langle\langle r\rangle\rangle\langle\langle r\rangle\rangle(\neg\textcircled{k} \wedge \langle\langle r\rangle\rangle(\neg\textcircled{k} \wedge \langle\langle r\rangle\rangle(\textcircled{k} \wedge \neg p)))\\
(\mathit{Tran}) & [\![r]\!](\neg p \rightarrow \langle\langle r\rangle\rangle(\textcircled{k} \wedge p \wedge [\![r]\!](\neg p \wedge \neg\textcircled{k} \rightarrow \langle\langle r\rangle\rangle(\textcircled{k} \wedge p \wedge\\
& \quad \langle\langle r\rangle\rangle(\textcircled{k} \wedge \neg p \wedge \langle\langle r\rangle\rangle(\textcircled{k} \wedge \neg p))))))).
\end{array}$$

Let *Inf* be $\mathit{Back} \wedge \mathit{Spy} \wedge \mathit{Irr} \wedge \mathit{Succ} \wedge \mathit{No\text{-}3cyc} \wedge \mathit{Tran}$, and let $\mathcal{M} = \langle D, \mathcal{I}, \emptyset\rangle$. The proof that $\mathcal{M}$ is infinite if $\mathcal{M}, w \models \mathit{Inf}$ is similar to the proof of Theorem 29 Instead of using i to identify the spy point we now use p and $\textcircled{k}$. $\textcircled{k}$ is needed to distinguish the spy point from other points where p might hold.

Notice that *Back*, *Spy*, *Succ*, and *No-3cyc* are very similar to the ones in the proof of Theorem 29, *Irr* forces $\mathcal{I}(r)$ to be irreflexive and *Tran* says that every pair of successors u and v are related (either $(u,v) \in \mathcal{I}(r)$ or $(v,u) \in \mathcal{I}(r)$), and this together with the other formulas, implies that $\mathcal{I}(r)$ is transitive. □

In a similar way, we can encode the $\omega \times \omega$ tiling problem to show that satisfiability in $\mathcal{ML}_\emptyset(\langle\langle r\rangle\rangle)$ is undecidable.

Theorem 32. *The satisfiability problem for $\mathcal{ML}_\emptyset(\langle\langle r\rangle\rangle)$ is undecidable.*

Proof. The formula φ^T needed for the encoding of a tiling problem T in this case is the conjunction of the following:

$$\begin{array}{rll}
(\mathit{Back}) & p \wedge [\![s]\!]\neg p \wedge \langle\langle s\rangle\rangle\top \wedge [\![s]\!]\langle\langle s\rangle\rangle(\textcircled{k} \wedge p) \wedge [\![s]\!][\![s]\!](\textcircled{k} \wedge p) & \\
(\mathit{Spy}) & [\![s]\!][\![\dagger]\!](\neg p \wedge \langle\langle s\rangle\rangle(\textcircled{k} \wedge p \wedge \langle\langle s\rangle\rangle(\textcircled{k} \wedge \neg\langle\langle\dagger\rangle\rangle\textcircled{k}))) & \dagger \in \{r,u\}\\
(\mathit{Grid}) & [\![s]\!]\langle\langle\dagger\rangle\rangle\top & \dagger \in \{r,u\}\\
(\mathit{Func}) & [\![s]\!][\![\dagger]\!]\langle\langle s\rangle\rangle\langle\langle s\rangle\rangle(\textcircled{k} \wedge \langle\langle\dagger\rangle\rangle\textcircled{k} \wedge [\![\dagger]\!]\textcircled{k}) & \dagger \in \{r,u\}\\
(\mathit{Conf}) & [\![s]\!]\langle\langle u\rangle\rangle\langle\langle r\rangle\rangle\langle\langle s\rangle\rangle\langle\langle s\rangle\rangle(\textcircled{k} \wedge \neg\langle\langle r\rangle\rangle\textcircled{k} \wedge \langle\langle u\rangle\rangle\textcircled{k} \wedge & \\
& \quad \langle\langle r\rangle\rangle\langle\langle u\rangle\rangle(\textcircled{k} \wedge \neg\langle\langle r\rangle\rangle\textcircled{k})) & \\
(\mathit{UR\text{-}no\text{-}Cycle}) & [\![s]\!][\![u]\!][\![r]\!]\neg\textcircled{k} \wedge [\![s]\!][\![r]\!][\![u]\!]\neg\textcircled{k} & \\
(\mathit{URU\text{-}no\text{-}Cycle}) & [\![s]\!][\![u]\!][\![r]\!][\![u]\!]\neg\textcircled{k} & \\
(\mathit{Unique}) & [\![s]\!]\left(\bigvee_{1\le i\le n} t_i \wedge \bigwedge_{1\le i<j\le n}(t_i \rightarrow \neg t_j)\right) & \\
(\mathit{Vert}) & [\![s]\!]\bigwedge_{1\le i\le n}\left(t_i \rightarrow \langle\langle u\rangle\rangle \bigvee_{1\le j\le n, u(T_i)=d(T_j)} t_j\right) & \\
(\mathit{Horiz}) & [\![s]\!]\bigwedge_{1\le i\le n}\left(t_i \rightarrow \langle\langle r\rangle\rangle \bigvee_{1\le j\le n, r(T_i)=l(T_j)} t_j\right). &
\end{array}$$

□

From the undecidability of $\mathcal{ML}_\emptyset(\langle\langle r\rangle\rangle)$, we can easily conclude the undecidability of $\mathcal{ML}(\langle r\rangle)$ and $\mathcal{ML}_\emptyset(\langle r\rangle)$.

Theorem 33. *$\mathcal{ML}_\emptyset(\langle r\rangle)$ lacks the finite model property and it is undecidable.*

Proof. Straightforward from Theorems 8, 31 and 32 □

To prove failure of the finite model property for the case $\mathcal{ML}(\langle r\rangle)$ we first notice that the following lemma is easy to establish (we only state it for the mono-modal case; a similar result is true in the multimodal case). Failure of the finite model property is then a direct consequence.

Lemma 34. *Let d be the modal depth of φ. If $\langle D, \mathcal{I}, M\rangle, w \models \left(\bigwedge_{i=0}^{d}[r]^i\neg\textcircled{k}\right) \wedge \varphi$ then $\langle D, \mathcal{I}, \emptyset\rangle, w \models \varphi$.*

Corollary 35. *$\mathcal{ML}(\langle r\rangle)$ lacks the finite model property.*

Proof. Using Lemma 34 we can prove that the formula $\left(\bigwedge_{i=0}^{4}[r]^i\neg\textcircled{k}\right) \wedge \mathit{Inf}$, where Inf is the formula in the proof of Theorem 31, forces an infinite model. □

Corollary 36. *The satisfiability problem for $\mathcal{ML}(\langle r\rangle)$ is undecidable.*

Proof. Using the idea of Lemma 34 and the formula φ^T in the proof of Theorem 32, we can obtain a formula ψ such that if $\mathcal{M}, w \models \psi$ then $\mathcal{M}$ is a tiling of $\omega\times\omega$. For the converse, we can build exactly the same model as in the proof of Theorem 32 and check that it satisfies ψ. □

Now we want to briefly mention the case of erase with respect to decidability. Given that Ⓔ can be seen as an operator that internalizes the notion of starting the evaluation of a formula with an empty memory, it is quite easy to establish the following result:

Theorem 37. *The satisfiability problem for $\mathcal{ML}(\langle\langle r\rangle\rangle, \textcircled{e})$ is undecidable. Furthermore, the logic lacks the finite model property.*

Proof. Theorems 31 and 32 show that the logic $\mathcal{ML}_\emptyset(\langle\langle r\rangle\rangle)$ lacks the finite model property and that its satisfiability problem is undecidable. It is straightforward to see that just adding Ⓔ in front of each encoding is enough to achieve the same results for $\mathcal{ML}(\langle\langle r\rangle\rangle, \textcircled{e})$. □

6 Conclusions and Further Work

In this article we investigated several memory logics. These logics were inspired by the hybrid logic $\mathcal{HL}(\downarrow)$ considering the $\downarrow$ operator as a storage command and the assignment function as a storage structure. The aim of this article is to explore this idea, investigating different ways in which information can be stored and retrieved, and the logics that result.

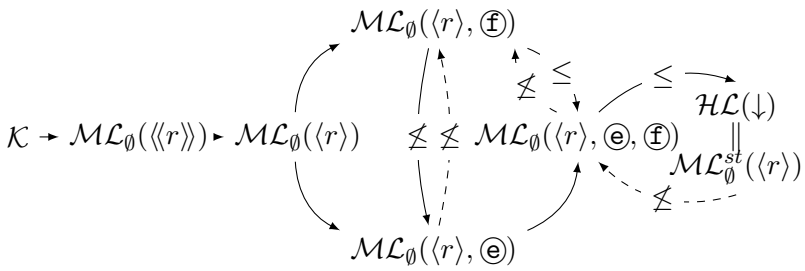


Figure 1: Different expressive power of memory logics

There are different dimensions in which the idea can be carried further, which we inspect in this article. We can, for example, vary the *type* of storage structure. An assignment function is a very sophisticated memory structure: it has unbounded size, it provides direct access to all its memory cells, and each stored element can be unequivocally retrieved. We discussed in detail the result of using a set (instead of a function) as the information container, and show that this change results in logics with strictly lower expressive power. We also show that if we replace the set with a richer structure that allows the unique identification of the elements stored (like is the case in a stack) we regain the full expressive power of $\mathcal{HL}(\downarrow)$.

The second dimension we analyzed was the collection of memory operators included in the language. We show that operators to add, test membership and delete elements from the memory can be naturally defined, and we mapped out the expressive power of the resulting logics.

Finally, the third dimension we investigated was the effect of imposing conditions on the state of the initial memory of the model in which we evaluate a formula. Requiring the initial memory to be empty (a natural requirement when working with models with state) boost the expressive power of the logic.

In terms of expressive power, the memory logics we presented lie between the basic modal logic $\mathcal{K}$ and the hybrid logic $\mathcal{HL}(\downarrow)$. Figure 1 summarizes the results established in this article. The solid unlabeled arrows represent the $<$ relationship, i.e., $\mathcal{L} \to \mathcal{L}'$ means that the logic $\mathcal{L}$ is strictly less expressive than the logic $\mathcal{L}'$. In some cases we specifically indicate other relations (like $\leq$ or $\not\leq$), and the dashed arrows show the suspected answers to the open questions 1 and 2 we formulated.

We also discussed in detail complexity results. In most cases the satisfiability problem of the languages we introduced is undecidable. We were able to pin down only two logics, $\mathcal{ML}(\langle\langle r\rangle\rangle)$ and $\mathcal{ML}(\langle\langle r\rangle\rangle, \textcircled{f})$, which still have the bounded tree model property, and established that their satisfiability problem is PSPACE-complete. In other words, to regain decidability we had to allow models with a potentially non empty memory and imposed (through the operator $\langle\langle r\rangle\rangle$) a very restricted policy to memorize states. To obtain these results we defined in Section 5 different equivalence preserving translations which can be used to transfer known results, for example, from $\mathcal{HL}(\downarrow)$ to $\mathcal{ML}(\langle r\rangle)$ and $\mathcal{ML}_\emptyset(\langle r\rangle)$. For

instance, both logics are compact and their formulas are preserved by generated sub-models (see Areces et al. (2001)).

The study we carried out in this paper draws a more detailed picture of the properties of of memory logics. We have investigated these logics in a number of recent papers (Areces, 2007; Areces, Figueira, & Mera, 2009; Areces, Figueira, Gorín, & Mera, 2009; Mera, 2009) in which we present complete axiomatizations, tableaux calculi, complexity analisys for model checking, and preliminary results on the Beth and the interpolation properties for different fragments of this family. But there is still work to be done.

Even though we obtained logics less expressive than $\mathcal{HL}(\downarrow)$, most of the logics we analyzed are undecidable. One of the motivations behind memory logics was to find decidable but yet useful logics to model scenarios with state. With this goal in mind several directions for future research suggest themselves. One possibility is to study memory logics as temporal logics, restricting the class of models to linear or tree structures which has shown to reduce complexity for hybrid logics (see (Areces et al., 2000)). The complexity of hybrid logics over restricted frame classes was investigated in detail by Schneider (2007), and a similar approach can be pursued for memory logics. In a related line, the freeze operator of Henzinger (1990) (a binding operator weaker than $\downarrow$ that bind *values* associated to the states instead of the states themselves) can also be further weakened using the ideas presented in this article. Finally decidable memory logics could also be obtained by following the approach of Meier et al. (2009) which instead of restricting the class of frames, imposed restrictions on the Boolean operators allowed in the language.

Acknowledgements: S. Figueira is partially supported by CONICET (grant PIP 370).

References

Alur, R., Courcoubetis, C., & Dill, D. (1993). Model-checking in dense real-time. *Information and Computation* **104**(1), 2–34.

Alur, R., Feder, T., & Henzinger, T. (1996). The benefits of relaxing punctuality. *Journal of the ACM* **43**(1), 116–146.

Alur, R., & Henzinger, T. (1989). A really temporal logic. In *Journal of the ACM*, pp. 164–169. IEEE Computer Society Press.

Areces, C. (2007). Hybrid logics: The old and the new. In Arrazola, X. & Larrazabal, J., editors, *Proceedings of LogKCA-07*, San Sebastian, Spain, pp. 15–29.

Areces, C., Blackburn, P., & Marx, M. (2000). The computational complexity of hybrid temporal logics. *Logic Journal of the IGPL* **8**(5), 653–679.

Areces, C., Blackburn, P., & Marx, M. (2001). Hybrid logics: characterization, interpolation and complexity. *The Journal of Symbolic Logic* **66**(3), 977–1010.

Areces, C., Figueira, D., Figueira, S., & Mera, S. (2008). Expressive power and decidability for memory logics. In *Logic, Language, Information and Computation*, Volume 5110 of *Lecture Notes in Computer Science*, pp. 56–68. Springer Berlin / Heidelberg. Proceedings of WoLLIC 2008.

Areces, C., Figueira, D., Gorín, D., & Mera, S. (2009). Tableaux and model checking for memory logics. In *Automated Reasoning with Analytic Tableaux and Related Methods*, Volume 5607 of *LNAI*, Oslo, Norway, pp. 47–61. Springer Berling / Heidelberg. Proceedings of Tableaux09.

Areces, C., Figueira, S., & Mera, S. (2009). Completeness results for memory logics. In *Proceedings of LFCS 2009*, Volume 5407 of *LNCS*, pp. 16–30. Springer.

Areces, C., & ten Cate, B. (2006). Hybrid logics. In Blackburn, P., Wolter, F., & van Benthem, J., editors, *Handbook of Modal Logics*, pp. 821–868. Elsevier.

Berman, F., & Paterson, M. (1981). Propositional dynamic logic is weaker without tests. *Theoretical Computer Science* **16**, 321–328.

Blackburn, P., de Rijke, M., & Venema, Y. (2001). *Modal Logic*. Cambridge University Press.

Blackburn, P., & Seligman, J. (1995). Hybrid languages. *Journal of Logic, Language and Information* **4**, 251–272.

Blackburn, P., Wolter, F., & van Benthem, J., editors (2006). *Handbook of Modal Logics*. Elsevier.

Börger, E., Grädel, E., & Gurevich, Y. (1997). *The classical decision problem*. Springer Verlag.

Ebbinghaus, H., Flum, J., & Thomas, W. (1984). *Mathematical Logic*. Springer-Verlag.

Floyd, R. (1967). Assigning meanings to programs. In *Proceedings of the American Mathematical Society Symposia on Applied Mathematics, Volume 19*, pp. 19–31.

Gerbrandy, J. (1999). *Bisimulations on Planet Kripke*. Ph. D. thesis, University of Amsterdam. ILLC Dissertation series DS-1999-01.

Groenendijk, J., & Stokhof, M. (1991a). Dynamic predicate logic. *Linguistics and Philosophy* **14**, 39–100.

Groenendijk, J., & Stokhof, M. (1991b). Two theories of dynamic semantics. In van Eijck, J., editor, *Logics in AI — European Workshop JELIA'90*, Lecture Notes in Artificial Intelligence, pp. 55–64. Springer-Verlag.

Harel, D. (1984). Dynamic logic. In Gabbay, D. & Guenthner, F., editors, *Handbook of Philosophical Logic. Vol. II*, Volume 165 of *Synthese Library*, pp. 497–604. Dordrecht: D. Reidel Publishing Co. Extensions of classical logic.

Harel, E., Lichtenstein, O., & Pnueli, A. (1990). Explicit clock temporal logic. In *Proceedings of LICS'90*, pp. 402–413.

Henzinger, T. (1990). Half-order modal logic: How to prove real-time properties. In *Proceedings of the Ninth Annual Symposium on Principles of Distributed Computing*, pp. 281–296. ACM Press.

Hoare, C. (1969). An axiomatic basis for computer programming. *Communications of the ACM* **12**(10), 576–583.

Koymans, R. (1990). Specifying real-time properties with metric temporal logic. *Real-Time Systems* **2**(4), 255–299.

Meier, A., Mundhenk, M., Schneider, T., Thomas, M., Weber, V., & Weiss, F. (2009). The complexity of satisfiability for fragments of hybrid logic—Part I. In *Proceedings 34th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, Volume 5734 of *LNCS*, pp. 587–599.

Mera, S. (2009). *Modal Memory Logics*. Ph. D. thesis, Universidad de Buenos Aires and Université Henri Poincaré.

Ouaknine, J., & Worrell, J. (2005). On the decidability of metric temporal logic. In *Proceedings of the 20th IEEE Symposium of Logic in Computer Science (LICS 2005)*, Chicago, IL, USA, pp. 188–197. IEEE Comp. Soc. Press.

Plaza, J. (1989). Logics of public communications. In *4th International Symposium on Methodologies for Intelligent Systems*, pp. 201–216.

Schneider, T. (2007). *The Complexity of Hybrid Logics over Restricted Classes of Frames*. Ph. D. thesis, University of Jena.

van Benthem, J. (2001). Logics for information update. In *TARK'01: Proceedings of the 8th Conference on Theoretical Aspects of Rationality and Knowledge*, pp. 51–67. Morgan Kaufmann Publishers Inc.

van Benthem, J. (2005). An essay on sabotage and obstruction. In *Mechanizing Mathematical Reasoning*, pp. 268–276.

van Benthem, J., van Eijck, J., & Kooi, B. (2006). Logics of communication and change. *Information and Computation* **204**(11), 1620–1662.

van Ditmarsch, H., van der Hoek, W., & Kooi, B. (2007). *Dynamic Epistemic Logic*. Kluwer academic publishers.